

Komplexe Forschungsmethoden

Gerrit Hirschfeld

Inhaltsverzeichnis

	Herzlich Willkommen	5
1	Einführung	7
1.1	Was Sie lernen werden	7
1.2	Wie dieses Buch aufgebaut ist	8
1.3	Was nicht Inhalt dieses Buches ist	8
1.4	Konzeptuelle Voraussetzungen	9
I	Querschnittsthemen	
2	Abbildungen	17
2.1	Warum Abbildungen wichtig sind	17
2.2	Wie man gute Abbildungen erstellt	20
2.3	Die Grammar of Graphics und ggplot2	25
2.4	Fazit	39
3	Datenmanagement	41
3.1	Folgen für das Datenmanagement	41
3.2	Tidy Data	43
3.3	Datenqualität und auffällige Daten	44
3.4	Datenmanagement mit R	47
3.5	Die einzelnen Schritte des Datenmanagements mit R	49
3.6	Fazit	63
II	ALM und Erweiterungen	
4	ALM und Multiple lineare Regression	67
4.1	Das Allgemeine Lineare Modell	67
4.2	Wie die Parameter geschätzt werden	73
4.3	Varianzaufklärung	76
4.4	Annahmen des linearen Modells	79
4.5	Regression mit R	81
4.6	Schicke Tabellen mit modelsummary	93
4.7	Fazit	99
5	Varianzanalysen	101
5.1	Die Idee der Varianzzerlegung	101
5.2	Mehrfaktorielle Varianzanalyse und Interaktionen	103

5.3	Kovarianzanalyse (ANCOVA)	105
5.4	Varianzanalyse mit Messwiederholung	106
5.5	Prä-post-Designs mit zwei Gruppen	107
5.6	Fallzahlplanung	109
5.7	Varianzanalysen mit R	110
5.8	Fazit	124
6	Logistische Regression	125
7	Das LASSO und andere Shrinkage-Verfahren	127
8	Mehrebenenmodelle	129
III Faktorenanalysen und Strukturgleichungsmodelle		
9	Exploratorische Faktorenanalyse	133
10	Konfirmatorische Faktorenanalyse	135
11	Messinvarianz	137
12	Veränderung modellieren	139
IV Einzelne Verfahren		
13	Conjoint-Analyse	143
	References	145
	Anhang	151
A	Erste Schritte mit R und RStudio	153
A.1	Was ist R und R Studio	153
A.2	R und RStudio installieren	153
A.3	Die RStudio-Oberfläche	154
A.4	R als Taschenrechner	154
A.5	Objekte und Zuweisungen	155
A.6	Pakete installieren und laden	155
A.7	Hilfe bekommen	156
A.8	Mit KI arbeiten	156

Herzlich Willkommen

Dieses Buch ist als Skript zur Vorlesung **Komplexe Forschungsmethoden der Wirtschaftspsychologie** im MSc. Studiengang Wirtschaftspsychologie an der Hochschule Bielefeld entstanden. Es richtet sich aber im Prinzip an alle, die etwas über menschliches Erleben und Verhalten herauszufinden möchten und dafür die aktuell am besten geeigneten Methoden verwenden wollen.

Die Analyse von Daten wird dabei eine wichtige Rolle einnehmen. Es ist aber keine allgemeine Einführung in Datascience - von denen es viele hervorragende gibt. Werden wir uns nicht nur damit beschäftigen, wie man gegebene **Daten effizient ausgewertet**, sondern auch wie man **Studien plant** und welche besonderen Herausforderungen **Psychologische Forschung** mit sich bringt.

Konkret deckt dieses Buch den klassischen Kanon der Methoden in der Psychologie - Multiple Regressionen, Varianzanalysen, explorative und konfirmatorische Faktorenanalysen Conjointanalysen ab. Einige Methoden davon haben Sie ggf., bereits kennengelernt, wir werden aber versuchen Sie alle auf denselben top aktuellen Stand zu bringen der es Ihnen ermöglicht diese Methoden auch einzusetzen. Außerdem werden mit der Visualisierung und Aufbereitung von Daten und dem Einsatz von LLMs zur Textanalyse auch Themen behandelt, die für die praktische Arbeit sehr nützlich sind und es Ihnen ermöglichen soll zu glänzen.

Die einzelnen Kapitel sind zweigeteilt aufgebaut. Der erste Teil gibt jeweils eine konzeptuelle Einleitung in das Thema und beschreibt häufige Probleme. Der zweite Teil beschreibt, wie man diese Schritte mit Hilfe von R umsetzen kann. Damit Sie das auch ausprobieren können, gibt es neben dem Fließtext ein sog. Workbook, in dem Sie direkt einzelne Code-Schnipsel ausprobieren können. Zur Vertiefung werden auch Projekte dargestellt, die Sie nutzen können, um mal alle Schritte von der Versuchsplanung bis zur Analyse durchzuführen.

Viel Spaß Gerrit Hirschfeld

P.s. die Anhänge zur Matrix-Algebra sind nicht Prüfungsrelevant. Ich wollte aber auch mal ein Angebot für diejenigen machen, die Spaß an Mathe haben.

1. Einführung

„I suppose it is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail.“ – (Maslow, 1962)

Die Psychologische Methodenlehre ist eine spannende und sehr dynamische Disziplin innerhalb der Psychologie. Das Ziel dieses Kurses ist Sie mit den wichtigsten Konzepten und Hintergrundinformationen zu versorgen und Ihnen beizubringen, wie Sie diese Methoden effektiv und effizient einsetzen können, um Forschungsfragen in der Psychologie zu beantworten. Nachdem Sie dieses Buch gelesen haben, werden Sie ganz viele unterschiedliche Untersuchungsmethoden kennengelernt haben. Die Welt wird hoffentlich nicht mehr aus nur aus Nägeln bestehen.

1.1 - Was Sie lernen werden

Empirische Forschung in der Psychologie beginnt nicht mit Daten, sondern mit einem Problem. Aus ihm entwickeln wir – gestützt auf Theorie und Forschungsstand – Hypothesen, daraus ein Untersuchungsdesign, eine Operationalisierung und eine Stichprobe; erst dann werden Daten erhoben, ausgewertet und interpretiert, und am Ende steht ein Bericht, der neue Fragen aufwirft (Abbildung 1.1). Obwohl dieser Ablauf eine gute Orientierung liefert, ist es wichtig zu bedenken, dass die einzelnen Phasen stark aufeinander bezogen sind. Die entscheidende Weichenstellen dafür welche Methode eingesetzt werden können fällt daher nicht erst bei der Analyse, sondern bei der Wahl und Planung des **Untersuchungsdesigns**. Umgekehrt ist es auch so, dass das Wissen um Auswertungsmethoden : Erst wer die Verfahren kennt, kann überhaupt die *richtigen* Hypothesen formulieren. Wer nur den t-Test kennt, plant eben Zwei-Gruppen-Experimente und stellt Zwei-Gruppen-Fragen. Ein breiteres Methodenrepertoire erlaubt bessere Fragen und bessere Studien – deshalb setzt dieses Buch für viele Verfahren bewusst früher an als bei der reinen Datenanalyse.

„To consult the statistician after an experiment is finished is often merely to ask him to conduct a post mortem examination. He can perhaps say what the experiment died of.“ – Ronald A. Fisher (1938)



Abbildung 1.1: Der empirische Forschungsprozess in der Psychologie

In diesem Buch geht es um komplexe Forschungsmethoden lernen. Damit sind in erster Linie Forschungsmethoden gemeint, die sich dadurch auszeichnen, dass sie **Viele Variablen gleichzeitig** betrachten können. Einfache Verfahren wie der t-Test betrachten wenige Variablen, multivariate Verfahren viele auf einmal. Dadurch können auch komplexere Untersuchungsdesigns, bei denen z.B. die einzelnen Beobachtungen nicht alle unabhängig voneinander sind, besser analysiert werden. Es gibt aber noch zwei weitere Aspekte, für die sich ein Buch in komplexen Forschungsmethoden lohnt. Erstens, wenn man nicht nur die Analyse der Ergebnisse in den Blick nimmt, sondern auch

Untersuchungen plant. Zweitens, indem wir die Methoden selbst in den Blick nehmen und Methoden kennenlernen um diese **zu bewerten und weiterzuentwickeln**.

Bei allen Methoden mit denen wir uns befassen, ist mir wichtig, dass Sie die ganzen Schritte des Forschungsprozesses so einer Studie verstehen, kritisieren und selbst einsetzen können. Das bedeutet, dass Sie einerseits die mathematischen Grundlagen verstehen, andererseits aber die Übersicht nicht verlieren und Ergebnisse jederzeit mit einfacheren, oft graphischen Methoden gegenprüfen können.

1.2 - Wie dieses Buch aufgebaut ist

Wir hangeln uns an den Methoden entlang. Zu jeder Methode gibt es einen gewissen mathematischen Hintergrund — das ist ein Zugang, um zu verstehen, was ein Verfahren eigentlich tut. Ein zweiter Zugang sind Simulationsstudien: Statt ein Verfahren nur herzuleiten, lassen wir es auf Daten mit bekannten Eigenschaften los und schauen, ob es sich so verhält, wie wir es erwarten.

Dieses Buch ist statisch. Das heißt: Sie können es ausdrucken und in der Badewanne oder am Strand lesen, ohne Angst zu haben, dass etwas kaputtgeht. Parallel gibt es ein Workbook mit interaktiven Übungen zu den einzelnen Kapiteln.

- **Datenmanagement und Visualisierung.** Bevor wir mit einzelnen Verfahren beginnen, besprechen wir zwei Themen, die einen Großteil der praktischen Arbeit ausmachen: das Datenmanagement (also das Importieren, Bereinigen und Transformieren von Daten) und die Visualisierung von Daten. Das Datenmanagement nimmt in der Praxis den größten Teil der Auswertungsarbeit am Computer in Anspruch; die Visualisierung wird in der Psychologie leider oft vernachlässigt.
- **Das Allgemeine Lineare Modell und seine Erweiterungen.** Ausgehend von der multiplen Regression erarbeiten wir die zugrunde liegende Idee des Allgemeinen Linearen Modells und wichtige Erweiterungen. Dazu gehören Mehrebenenmodelle, mit denen sich auch komplexe Abhängigkeiten zwischen Beobachtungen modellieren lassen, sowie Varianten für andere Datentypen (z. B. die logistische Regression) und regularisierte Verfahren (z. B. die LASSO-Regression).
- **Faktorenanalysen und Strukturgleichungsmodelle.** Anschließend sprechen wir über Faktorenanalysen und Strukturgleichungsmodelle, mit denen sich die Beziehungen zwischen mehreren (auch latenten) Variablen untersuchen lassen.
- **Item-Response-Theorie.** Ein eigenes Kapitel widmet sich der Item-Response-Theorie als maßgeblicher Grundlage für die Entwicklung von (adaptiven) Tests.
- **Clusteranalyse.** Danach behandeln wir die Clusteranalyse, mit der sich Gruppen von Fällen datengetrieben identifizieren lassen.
- **LLMs** Zum Schluss lernen Sie Methoden kennen, um effizient mit Sprachmodellen sog. LLMs zu arbeiten.

1.3 - Was nicht Inhalt dieses Buches ist

Ein Buch wird auch durch das definiert, was es bewusst weglässt. Zwei Dinge grenze ich ausdrücklich aus.

Bayesianische Statistik. Neben der in diesem Buch verwendeten *frequentistischen* Statistik gibt es eine lange Tradition, die das eigentliche Problem in der Logik des p-Werts selbst verortet und stattdessen für **bayesianische Verfahren** wirbt (Wagenmakers et al., 2017). Diese quantifizieren Evidenz direkt (etwa über Bayes-Faktoren) und beziehen Vorwissen (Priors) systematisch ein. Der Ansatz ist elegant und in manchen Fragestellungen klar überlegen. In diesem Buch spielt er dennoch nur eine untergeordnete Rolle – aus zwei praktischen Gründen: Erstens ist die überwältigende Mehrheit der Verfahren und Studien, die Ihnen begegnen werden, frequentistisch; Sie müssen sie lesen und verstehen können. Zweitens sind die wichtigsten Lehren der letzten Jahrzehnte – ausreichende Power, Schätzen statt bloßem Testen, Transparenz – weitgehend *unabhängig* davon, ob man frequentistisch oder bayesianisch rechnet. Wir bleiben daher beim frequentistischen Rüstzeug und greifen bayesianische Methoden nur dort auf, wo sie besonders hilfreich sind.

Reine Data Science und Programmierung. Sie werden auch kein Profi in Data Science. Es wird weiterhin Menschen geben, die schneller Daten hin- und herschieben und bessere Werkzeuge dafür haben. Unser Ziel ist ein anderes: Sie sollen nicht nur gegebene Daten auswerten, sondern auch überlegen und planen können, welche Daten besonders relevant für die Beantwortung einer Fragestellung aus der Praxis ist.

1.4 - Konzeptuelle Voraussetzungen

Da Sie eine Vorlesung in *komplexen* Forschungsmethoden der Psychologie besuchen, haben Sie mit hoher Wahrscheinlichkeit bereits mehrere Kurse in nicht-komplexen Forschungsmethoden erfolgreich absolviert. In einem typischen (wirtschafts-) Psychologie sind das Kurse in Methodenlehre (gerne I und II), Statistik, Diagnostik und Experimentalpraktika absolviert und selbst eine empirische Abschlussarbeit geschrieben. Bevor wir mit einzelnen Verfahren beginnen, möchte ich dennoch einige Grundgedanken wiederholen, die erklären, warum wir in der Psychologie bestimmte Schritte gehen und andere nicht.

1.4.1 - p-Werte und ihre Kritik

Das vertrauteste Werkzeug der psychologischen Statistik ist der **Nullhypothesen-Signifikanztest**, und in seinem Zentrum steht der **p-Wert**. Ronald Fisher prägte die Idee, ein Ergebnis als „signifikant“, zu bezeichnen, wenn es unter der Nullhypothese hinreichend unwahrscheinlich ist; ergänzt um die Alpha- und Beta-Fehler von Neyman und Pearson wurde daraus das Standardwerkzeug der Disziplin. Sein Reiz liegt auch darin, dass er gut zum kritischen Rationalismus **Karl Poppers** (2007) passt: Statt eine Theorie zu „beweisen“, versuchen wir, eine Nullhypothese zu *widerlegen*. Diese Nähe zur Falsifikation ist attraktiv – auch wenn die statistische Nullhypothese selten mit der eigentlich interessierenden Theorie identisch ist.

An diesem Werkzeug hat sich über Jahrzehnte eine tiefgreifende Kritik entzündet, die man kennen muss, um die heutige Methodenpraxis zu verstehen.

Das Power-Problem. Schon Cohen (1988) zeigte, dass viele psychologische Studien viel zu kleine Stichproben hatten, um selbst mittelgroße Effekte zuverlässig zu finden – ihre statistische **Power** war gering. Das Beunruhigende daran: Bei geringer Power sagt ein „nicht signifikantes“, Ergebnis fast nichts aus, und ein „signifikantes“ ist überraschend oft ein Fehlalarm. Cohen und andere warben jahrzehntelang

Statistischer Test. Ein Verfahren, das anhand von Daten zwischen zwei Hypothesen entscheidet – meist einer Nullhypothese und einer Alternativhypothese.

für Poweranalysen und für die Berichterstattung von **Effektstärken** — mit erstaunlich wenig Wirkung: Sedlmeier & Gigerenzer (1989) stellten fest, dass sich die Power der publizierten Studien über Jahre kaum verbessert hatte.

Die „New Statistics“. Als Reaktion auf die Fixierung auf das Sternchen hinter dem p-Wert plädierte Geoff. Cumming (2012) für eine *neue Statistik*: weg vom dichotomen „signifikant/nicht signifikant“, hin zur **Schätzung** von Effektstärken mit **Konfidenzintervallen** und zur kumulativen Bündelung von Befunden in Meta-Analysen. Nicht „Gibt es einen Effekt?“, sondern „Wie groß ist er, und wie sicher sind wir uns?“ sollte die Leitfrage sein.

Die Replikationskrise. Um 2011 kippte die Stimmung. Auslöser war unter anderem eine Studie von Daryl Bem (2011), der damals gängige Erhebungs- und Auswertungsmethoden verwendete, und mit ihnen zu dem Schluss kam, dass Probanden tatsächlich zu übernatürlichen Fähigkeiten wie dem Vorraussagen der nächsten Karte in einem gemischten Kartendeck in der Lage sind. Eine Reihe von Arbeiten haben dann herausgearbeitet welche sog. Freiheitsgrade (aka researcher degrees of freedom) dafür verantwortlich sind, dass man es liegt, dass man praktisch *jedes* Ergebnis signifikant bekommt Simmons et al. (2011). Den empirischen Höhepunkt lieferte die von Brian Nosek koordinierte **Open Science Collaboration** (Open Science Collaboration, 2015): Von 100 wiederholten Studien ließ sich nur etwa ein Drittel bis die Hälfte replizieren. Das war der Beginn dessen, was heute als **Replikationskrise** bekannt ist. Wichtig ist zu betonen, dass auch in anderen Feldern in denen es untersucht wurde, die Rate an replizierbaren Ergebnissen deutlich geringer ausfällt. So hat sich in der Krebsforschung herausgestellt, dass von 53 Landmark Befunden aus der präklinischen Krebsforschung nur 6 (11%) replizieren ließen (Begley & Ellis, 2012).

Metascience. Diese Krise hatte einen überaus produktiven Effekt: Sie machte die Forschungspraxis selbst zum Forschungsgegenstand. Aus der Reformbewegung gingen **Präregistrierung**, **Registered Reports**, offene Daten und groß angelegte Replikationsprojekte hervor (Munafò et al., 2017). Damit ist die Psychologie zu einem Vorreiter der **Metascience** geworden — der wissenschaftlichen Untersuchung des Wissenschaftsbetriebs selbst. Studien in Metascience beschäftigen sich damit welche Forschungsmethoden eingesetzt werden und wie sich dies über die Zeit verändert.

1.4.2 - Simulationsstudien als Forschungsmethoden

Neben der Frage welche Methoden in der Praxis eingesetzt werden, gibt es auch sehr viel Forschung dazu, welche Methoden eingesetzt werden sollten. Wenn man die Methoden zum Gegenstand der Forschung macht, stellt man sich die Frage wie gut funktioniert eine Methode eigentlich, und unter welchen Bedingungen?

- Hält ein t-Test sein Alpha-Niveau von 5 % ein, wenn die Varianzen in den Gruppen ungleich sind?
- Wie viel Power habe ich bei einer Stichprobe von nur 20 Personen pro Gruppe?
- Liefert ein Verfahren verzerrte Schätzungen, wenn Werte fehlen?

Solche Fragen lassen sich zum Teil durch mathematische Herleitungen beantworten, für realistische Szenarien wird das aber schnell unhandlich. Das wichtigste und zugleich zugänglichste Werkzeug der

angewandten Statistik ist daher die **Simulationsstudie** (T. P. Morris et al., 2019). Die Grundidee ist recht einfach (vgl. Abbildung 1.2):

1. Man erzeugt Daten mit **bekannten Eigenschaften** — man kennt also die „Wahrheit“, weil man sie selbst festgelegt hat.
2. Man wendet das interessierende Verfahren auf diese Daten an und protokolliert das Ergebnis (z. B. den p-Wert).
3. Man wiederholt die Schritte 1 und 2 sehr oft — typischerweise tausende Male — und wertet aus, ob sich das Verfahren so verhält, wie man es erwartet.

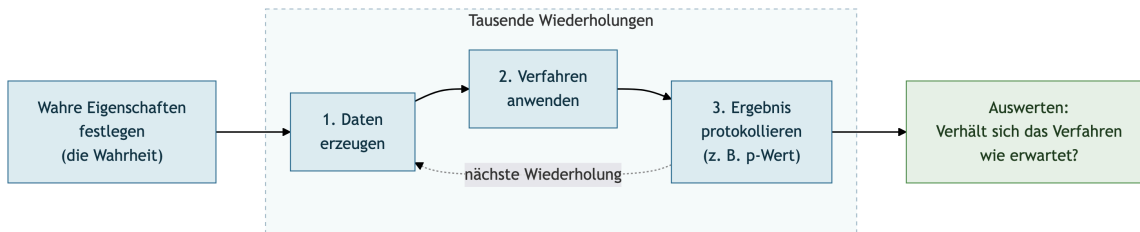


Abbildung 1.2: Die Logik einer Simulationsstudie

Ein kleines Beispiel um den Einfluss der Varianzhomogenität auf die Performanz von t-tests zu zeigen. Wir generieren Daten aus zwei Gruppen, zwischen denen es *keinen* echten Mittelwertsunterschied gibt, in der einen Gruppe sind 30 und in der anderen Gruppe 60 Personen. Zuerst simulieren wir die Daten so, dass beide Gruppen dieselbe Varianz haben. Mit den Daten rechnen wir dann einen t-Test und schreiben uns den resultierenden p-Wert auf. Das machen wir 10.000 mal. Wenn der Test hält, was er verspricht, sollten ungefähr 5 % der p-Werte kleiner als .05 sein — also etwa 500 von 10.000. Spannend wird es, wenn wir die erzeugten Daten gezielt manipulieren: Was passiert mit dieser Fehlerrate, wenn wir die Daten so generieren, dass die Varianzen in der kleineren Gruppe doppelt so hoch oder nur halb so hoch wie in der größeren Gruppe sind?

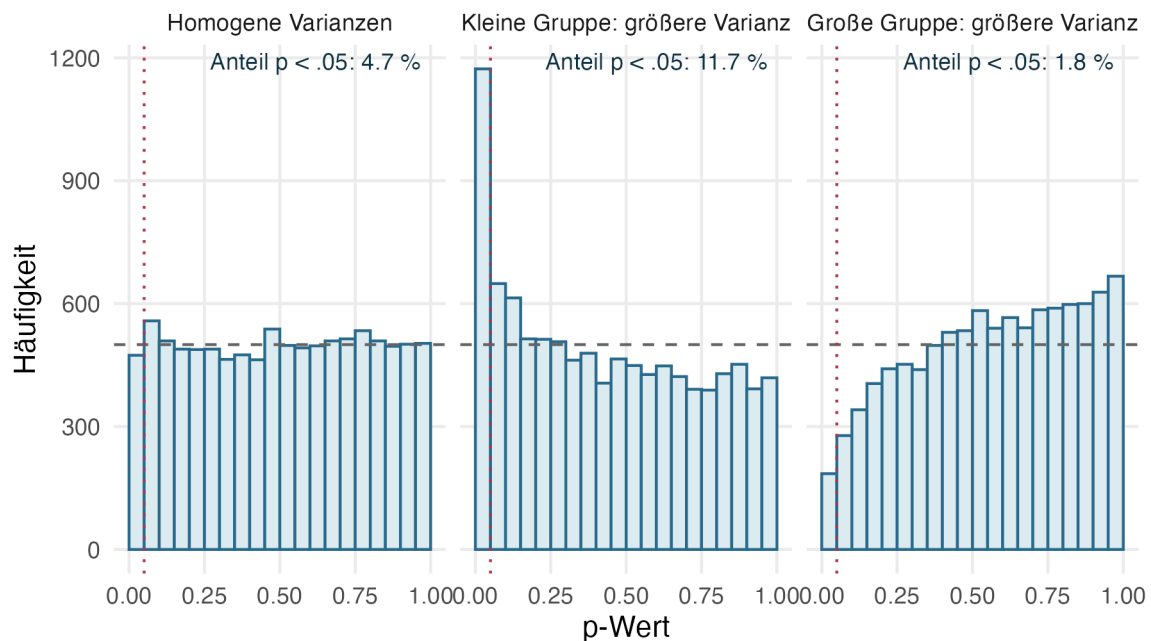


Abbildung 1.3: Verteilung der p-Werte aus je 10.000 t-Tests unter der Nullhypothese (kein wahrer Unterschied) bei ungleichen Gruppengrößen ($n = 30$ vs. $n = 60$). Die gestrichelte Linie markiert die bei einem korrekten Test erwartete Gleichverteilung, die gepunktete Linie die Schwelle $p = .05$.

Das Ergebnis Abbildung 1.3 ist eindrücklich: Bei homogenen Varianzen sind die p-Werte gleichverteilt, und die Fehlerrate liegt wie erwartet bei rund 5 %. Der Test funktioniert wie gewünscht. Sobald aber die *kleinere* Gruppe die größere Varianz hat, häufen sich kleine p-Werte, und der Test wird deutlich zu liberal (die Fehlerrate steigt klar über 5 %). Hat umgekehrt die *größere* Gruppe die größere Varianz, verschieben sich die p-Werte nach oben, und der Test wird zu konservativ. Die Faustregel „bei ungleichen Gruppen aufpassen“, bekommt so ein konkretes Gesicht – und man sieht sofort, warum der Welch-Test (in R die Voreinstellung) die bessere Wahl ist.

Simulationsstudien haben für uns einen doppelten Wert. Zum einen machen sie **Annahmen erfahrbar**: Statt den Satz „der t-Test setzt Varianzhomogenität voraus“, auswendig zu lernen, sehen Sie unmittelbar, was passiert, wenn diese Annahme verletzt ist. Zum anderen können Sie diese nutzen, um die Methoden zu überprüfen. Sie müssen einem neuen oder unbekannten Verfahren nicht blind vertrauen, sondern können sein Verhalten selbst untersuchen. Wir werden deshalb immer mal wieder kleinere Simulationsstudien nutzen, um zu checken, ob die Methoden auch das tun, was wir denken.

1.4.3 - Modelle: erklären oder vorhersagen?

„All models are wrong, but some are useful.“ – George Box (1976).

Fast alle Verfahren in diesem Buch beruhen auf **statistischen Modellen**. Ein Modell ist eine vereinfachte, formale Beschreibung, wie die beobachteten Daten zustande gekommen sein könnten. Wir tun so, als seien die Daten nach einer bestimmten, überschaubaren Regel entstanden, und schätzen dann die Größen (Parameter) dieser Regel. Weil ein Modell die Wirklichkeit immer vereinfacht, ist keines im strengen Sinn „richtig“. Für uns heißt das: Die Frage ist nie, ob ein Modell wahr ist, sondern ob es für unseren Zweck nützlich ist – und ob wir seine Grenzen kennen.

Was „nützlich,“ bedeutet, hängt allerdings vom Zweck ab. Shmueli (2010) unterscheidet zwei grundverschiedene Ziele:

- **Erklären.** Wir wollen verstehen, *warum* etwas passiert, und Hypothesen über Mechanismen prüfen. Hier zählt, ob die Modellstruktur der vermuteten (kausalen) Struktur entspricht; ein theoretisch begründetes, sparsames Modell ist oft aussagekräftiger als ein komplexes. Praktisch werden hierfür oft multiple regressionen verwendet, bei denen man schätzen kann, wie stark der „Einfluss“ unterschiedlicher Variablen auf das Outcome ist.
- **Vorhersagen.** Wir wollen für neue Fälle möglichst treffsichere Prognosen, unabhängig davon, ob wir den Mechanismus verstehen. Hier zählt allein die Vorhersagegüte an neuen Daten – auch ein „Black-Box“-Modell bei dem wir im Extremfall gar nicht sagen können woran es liegt, dass eine bestimmte aussage getroffen wurde, ist willkommen.

Beides ist legitim, aber die Entscheidungen unterscheiden sich: Ein Modell, das am besten erklärt, sagt nicht unbedingt am besten vorher – und umgekehrt. In der Psychologie liegt der Schwerpunkt traditionell beim Erklären; mit dem Einzug von Machine-Learning-Methoden gewinnt aber auch die Vorhersageperspektive an Bedeutung. Es hilft, sich bei jeder Analyse zu fragen, in welchem der beiden Modi man gerade arbeitet. Während der Schwerpunkt des Buches auf Methoden beruht, die eher zum Erklären passen, werden wir auch moderne Vorhersagemethoden (z.B. das LASSO) besprechen.

1.4.4 - Was Sie schon mitbringen sollten

Ein Bachelorstudium der Psychologie vermittelt eine ganze Reihe methodischer Grundlagen, auf die dieses Buch aufbaut. Sie sollten daher die folgenden Fragen beantworten können:

1. Was bedeutet es, eine Hypothese *konfirmatorisch* statt *exploratorisch* zu prüfen – und warum ist diese Unterscheidung so wichtig?
2. Was sind die besonderen Merkmale eines strengen Experiments (Manipulation, Kontrolle, Randomisierung)?
3. Worin unterscheiden sich experimentelle, quasi-experimentelle und korrelative Designs – und warum erlaubt nur das Experiment kausale Schlüsse?
4. Was bedeuten interne und externe Validität, und welche typischen Störgrößen (Confounder) gefährden sie?
5. Was ist der Unterschied zwischen einem Between- und einem Within-Design (Messwiederholung), und welche Folgen hat das für die Auswertung?
6. Was heißt Operationalisierung – wie wird aus einem theoretischen Konstrukt eine messbare Variable?

Messen und psychometrische Qualität

7. Welche Skalenniveaus gibt es, und welche Auswertungen erlauben sie jeweils?
8. Was sind die psychometrischen Gütekriterien (Objektivität, Reliabilität, Validität), wie misst man sie, und wie hängen sie zusammen?
9. Was besagt die Klassische Testtheorie, und was sagen Itemkennwerte wie Trennschärfe und interne Konsistenz über die Qualität einer Skala?

Beschreiben und Zusammenhänge

10. Welche Maße der zentralen Tendenz und der Streuung gibt es, wann verwendet man welches, und wozu dient die Standardisierung (z-Werte)?
11. Wie unterscheiden sich Kovarianz und Korrelation, wie interpretiert man einen Korrelationskoeffizienten, und hängt Korrelation mit Kausalität zusammen?

Testen, Schätzen und Standardverfahren

12. Was ist ein statistischer Test – Nullhypothese, Prüfgröße und p-Wert?
13. Wie hängen Alpha-Fehler, Beta-Fehler, Effektstärke und Stichprobenumfang (Power) zusammen – und wie unterscheidet sich statistische Signifikanz von Effektstärke und praktischer Relevanz?
14. Wie wählt man mithilfe eines Entscheidungsbaums das passende Standardverfahren (t-Test, χ^2 -Test, Korrelation, ANOVA, Regression) für eine Fragestellung?
15. Was versteht man unter Moderation und Mediation, und wie lassen sich diese Effekte statistisch prüfen?

Wo bekommt man dieses Wissen her? Am schnellsten aus den eigenen Bachelor-Unterlagen – die allermeisten dieser Punkte wurden dort behandelt. Zum Auffrischen und Vertiefen eignen sich die einschlägigen Lehrbücher: Für Untersuchungsdesign und Evaluation ist das Buch von Bortz und Döring Döring (2023) der deutschsprachige Klassiker. Wer Forschungsmethoden und Statistik aus einem Guss sucht, ist mit dem Buch von Eid und Kollegen gut bedient Eid et al. (2017), das beide Bereiche verbindet. Die statistischen Grundlagen lassen sich zusätzlich im Buch von Bortz (2010) nachschlagen. Einen besonders zugänglichen und praxisnahen Einstieg bietet das Buch von Field (2026). Speziell für die psychometrischen Themen (Gütekriterien, Testtheorie) ist Moosbrugger und Kelava (2020) das Standardwerk.

Überschnittsthemen

2	Abbildungen	17
2.1	Warum Abbildungen wichtig sind	17
2.2	Wie man gute Abbildungen erstellt	20
2.3	Die Grammar of Graphics und ggplot2	25
2.4	Fazit	39
3	Datenmanagement	41
3.1	Folgen für das Datenmanagement	41
3.2	Tidy Data	43
3.3	Datenqualität und auffällige Daten	44
3.4	Datenmanagement mit R	47
3.5	Die einzelnen Schritte des Datenmanagements mit R	49
3.6	Fazit	63

2. Abbildungen

„...the disputes that for so many generations have vexed philosophers are destroyed by **visible certainty**, and we are liberated from wordy arguments.“ – Galileo Galilei (1610/2016)

In diesem Kapitel lernen Sie, warum Abbildungen für komplexe Datenanalysen unverzichtbar sind, welche konzeptuellen Prinzipien guten Abbildungen zugrunde liegen und wie Sie mit dem Paket `ggplot2` sehr effizient auch komplexe Abbildungen erstellen können. Die erste Hälfte des Kapitels ist bewusst konzeptuell: Sie soll Ihnen ein Gespür dafür geben, *wann* und *wozu* Sie Abbildungen einsetzen. Die zweite Hälfte ist praktisch und zeigt Schritt für Schritt, wie Sie in R zu publikationsreifen Grafiken kommen.

Galileo hatte recht: Gute Forschungsmethoden erzeugen häufig *visible certainty*. Während er aber von der Erfindung des Teleskops sprach, kann man dasselbe von modernen Methoden der Datenvisualisierung sagen. Diese macht Zusammenhänge unmittelbar sichtbar, die in einer Tabelle voller Zahlen verborgen bleiben. Abbildungen haben im Kontext von komplexen Forschungsmethoden unterschiedliche Funktionen, die man voneinander abgrenzen kann. Die augenscheinlichste Funktion ist es Analyseergebnisse effektiv an ein Publikum zu **kommunizieren**. Die zweite Funktion ist es Daten zu **explorieren**. Wir haben im letzten Kapitel bereits davon gesprochen, dass psychologische Forschung oft hypothesengetrieben ist, es ist aber auch wichtig schnell einzelne Ideen zu Zusammenhängen darstellen zu können. Drittens eignen sich Abbildungen ganz hervorragend um die Daten zu checken und Voraussetzungen zu **prüfen**. In angewandten Forschungsprojekten gibt es daher ein paar Grafiken, die man nur für sich selbst macht, andere, die man mit Kollegen bespricht und zuletzt solche, die man final publizieren lassen möchte.

2.1 - Warum Abbildungen wichtig sind

2.1.1 - Anscombes Quartett: Kennwerte können täuschen

Ein statistisches Modell ist immer nur so gut wie die Annahmen, auf denen es beruht. Eine lineare Regression etwa unterstellt einen linearen Zusammenhang, geht von annähernd normalverteilten Residuen aus und reagiert empfindlich auf einzelne Ausreißer. Das Tückische ist: Verletzungen dieser Annahmen schlagen sich häufig gerade *nicht* in den üblichen Kennwerten wie Mittelwert, Standardabweichung oder Korrelation nieder.

Ein sehr anschauliches Beispiel hierfür wurde von Francis Anscombe entwickelt, das sog. **Anscombes Quartett** (Anscombe, 1973). Es besteht aus vier Datenpaaren X und Y. Die Tabelle 2.1 fasst die zentralen Kennwerte der vier Datensätze zusammen.

Tabelle 2.1: Deskriptive Kennwerte der vier Datensätze aus Anscombes Quartett. Alle vier stimmen bis auf Rundungsfehler überein.

Datenpaar	M (x)	SD (x)	M (y)	SD (y)	r (x, y)
1	9	3.32	7.5	2.03	0.82
2	9	3.32	7.5	2.03	0.82
3	9	3.32	7.5	2.03	0.82
4	9	3.32	7.5	2.03	0.82

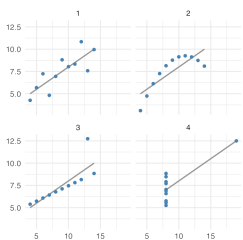


Abbildung 2.1: Anscombes Quartett: gleiche Kennwerte, verschiedene Struktur.

Die Mittelwerte, Standardabweichungen und Korrelationen sind praktisch identisch. Auch die Korrelation ist in allen vier Datenpaaren gleich bei $r = 0.82$, eine lineare Regression würde für alle vier Datensätze dieselbe Gerade schätzen. Ein Blick auf die zugehörige Abbildung 2.1 offenbart jedoch vier grundverschiedene Situationen:

Nur im ersten Datenpaar ist die Annahme eines linearen Zusammenhangs und damit auch die Interpretation eines positiven Zusammenhangs tatsächlich angemessen. Das zweite zeigt einen klar *nicht-linearen* (quadratischen) Verlauf. Solche Zusammenhänge finden sich immer wieder, wenn man den Zusammenhang zwischen Stress oder Arousal. Während eine Zunahme von Stress erstmal dazu führt, dass die Leistung steigt, werden diese Zuwächse immer geringer und kehren sich irgendwann um. Eine gute Erklärung für diesen Verlauf liefern Beerendonk und Kollegen (2024). Das ist auch für Manager wichtig, die denken, dass sie immer mehr Leistung aus Mitarbeitern herauspressen können. Das dritte Variablenpaar stellt einen perfekten linearen Zusammenhang dar, der durch einen einzigen Ausreißer nach unten verzerrt wird. Das vierte Datenpaar und insbesondere die Korrelation, die man daraus berechnen kann, beruht ganz auf einem einzigen einflussreichen Punkt.

Wichtig ist, dass diese vier nicht die einzigen Sonderfälle sind, die man sich vorstellen kann oder die in der Praxis auch vorkommen. Matejka und Fitzmaurice (2017) erzeugten mit dem *Datasaurus Dozen* noch viele weitere Datensätze die auch wieder identische numerische Zusammenfassungen erzeugen aber alle vollkommen unterschiedlich aussehen (Abbildung 2.2). Sie sollten sich also immer die Daten nicht nur deskriptiv als Tabelle sondern auch als Abbildungen anschauen.

„...make both calculations and graphs. Both sorts of output should be studied; each will contribute to understanding.“
— F. Anscombe (1973)

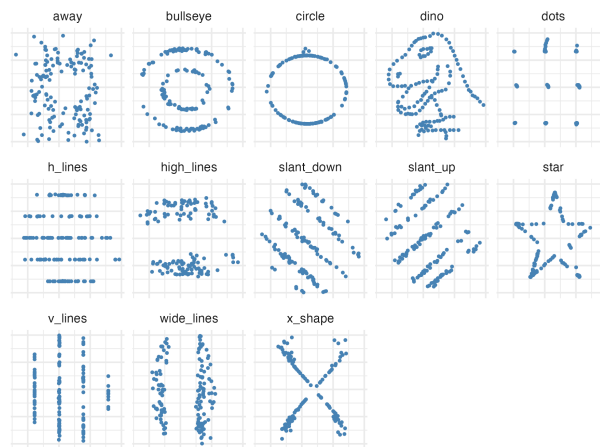


Abbildung 2.2: Der Datasaurus Dozen: dreizehn Datensätze mit praktisch identischen Kennwerten, aber völlig unterschiedlicher Gestalt (Matejka & Fitzmaurice, 2017).

2.1.2 - Explorative Datenanalyse

Die Idee, Daten nicht nur zur Bestätigung vorab formulierter Hypothesen, sondern auch zur *Entdeckung* von Struktur zu nutzen, geht auf John Tukey (1977) zurück. Er prägte den Begriff der **Explorativen Datenanalyse** (Exploratory Data Analysis, EDA) und stellte sie der *konfirmatorischen* Datenanalyse gegenüber.

Beide Modi verfolgen unterschiedliche Ziele:

- **Konfirmatorisch:** Sie testen eine vorab festgelegte Hypothese mit einem passenden statistischen Verfahren. Die Frage ist bereits gestellt, bevor Sie die Daten sehen.
- **Explorativ:** Sie befragen die Daten offen. Welche Verteilungen liegen vor? Gibt es unerwartete Gruppen, Zusammenhänge, Auffälligkeiten? Welche Hypothesen legen die Daten überhaupt nahe?

Abbildungen sind das zentrale Werkzeug der EDA. Ein Histogramm deckt Schiefe oder Mehrgipfligkeit auf, ein Streudiagramm zeigt nicht-lineare Zusammenhänge, ein Boxplot pro Gruppe macht unterschiedliche Streuungen sichtbar. Ganz praktisch dient die explorative Grafik auch der **Datenqualität**: Unmögliche Werte (ein Alter von 200 Jahren), fehlerhafte Kodierungen oder systematisch fehlende Werte fallen im Bild oft sofort auf, während sie in Kennwerten untergehen.

EDA unterscheidet zwischen *explorativer* Datenanalyse (Struktur entdecken, Hypothesen erzeugen) und *konfirmatorischer* Datenanalyse (vorab festgelegte Hypothesen prüfen).

⚠ Explorative Befunde sind Hypothesen, keine Ergebnisse

Die Stärke der EDA ist zugleich ihre Gefahr. Wer lange genug in einem Datensatz sucht, findet fast immer *irgendein* auffälliges Muster — auch dann, wenn in der Grundgesamtheit gar kein Effekt existiert. Ein rein explorativ „entdeckter“ Zusammenhang darf daher nicht ohne Weiteres als bestätigt gelten. Er ist eine *Hypothese*, die anschließend an neuen Daten konfirmatorisch geprüft werden muss. Diese Trennung ist eigentlich nur dann aufrecht zu erhalten, wenn man die Hypothesen vor der Datenerhebung präregistriert hat.

2.2 - Wie man gute Abbildungen erstellt

Nachdem geklärt ist, dass man immer Abbildungen erstellen sollte, stellt sich als nächste Frage wie man möglichst gute Abbildungen erstellt.

2.2.1 - Prinzipien guter Abbildungen

Wann ist eine Abbildung *gut*? Die einflussreichsten Antworten stammen von Edward Tufte, der in einer Reihe von Büchern (2013) sehr anschaulich und nachvollziehbar dargelegt hat, wie man Abbildungen nutzen kann, um besser über die dargestellten Sachverhalte nachdenken zu können und im besten Fall auch bessere Entscheidungen zu treffen. Für unsere Zwecke lassen sich seine Überlegungen auf einige wenige Prinzipien verdichten.

Show the data. Aggregierte Darstellungen (etwa reine Balken für Mittelwerte) verbergen die zugrunde liegende Verteilung. Wo es die Datenmenge erlaubt, sollten die Rohdaten sichtbar bleiben – etwa als Punkte, notfalls überlagert mit einer Zusammenfassung.

Grafische Integrität. Eine Abbildung darf nicht mehr behaupten, als die Daten hergeben. Tufte formalisiert das im *Lie Factor*: dem Verhältnis der im Bild dargestellten Effektgröße zur tatsächlichen Effektgröße in den Daten. Klassische Verstöße sind abgeschnittene y-Achsen, die kleine Unterschiede dramatisch wirken lassen, oder Flächen und Volumina, die eine eindimensionale Größe darstellen und den Effekt dadurch überzeichnen.

Data-Ink-Ratio. Ein möglichst großer Anteil der „Tinte“, einer Abbildung sollte tatsächlich Daten darstellen. Alles Übrige – dekorative Rahmen, Schlagschatten, Farbverläufe, überflüssige Gitterlinien – bezeichnet Tufte als **Chartjunk**. Die Faustregel lautet: Entfernen Sie alles, was Sie weglassen können, ohne dass Information verloren geht.

Datendichte und Small Multiples. Gute Abbildungen zeigen viel Information auf wenig Raum. Ein besonders wirkungsvolles Mittel sind *Small Multiples*: dieselbe Abbildung, viele Male für verschiedene Teilgruppen wiederholt und nebeneinander angeordnet. Das Auge erkennt Unterschiede zwischen kleinen, gleich aufgebauten Bildern mühelos – Anscombes Quartett oben ist genau ein solches Small-Multiple-Design.

Leider folgen viele der Abbildungen auch in wissenschaftlichen Fachzeitschriften nicht diesen Prinzipien sondern nutzen Abbildungen mit absolut minimalem Informationsgehalt. In der Abbildung 2.3 links sind wenn man ehrlich ist nur zwei Datenpunkte dargestellt, der Mittelwert vorher und der Mittelwert nachher. Es wird sehr viel Tinte und Platz darauf verwendet das darzustellen. Im Gegensatz dazu sind in der Abbildung rechts alle 60 Rohdatenpunkte dargestellt. Auch die gepaarte Natur der Daten wird explizit gemacht und Konfidenzintervalle¹ sowie ein Violin-Plot zur Darstellung der Dichteverteilung sind enthalten. Die Rohdaten helfen enorm dabei den Unterschied zu kontextualisieren, während die Verbindungslinien es ermöglichen sofort zu identifizieren, dass sich nur 3 von 30 Personen verschlechtert haben.

„Graphical excellence is that which gives to the viewer the greatest number of ideas in the shortest time with the least ink in the smallest space“
— E. Tufte 1983

¹Noch aussagekräftiger wäre hier sogar das Konfidenzintervall der Differenz.

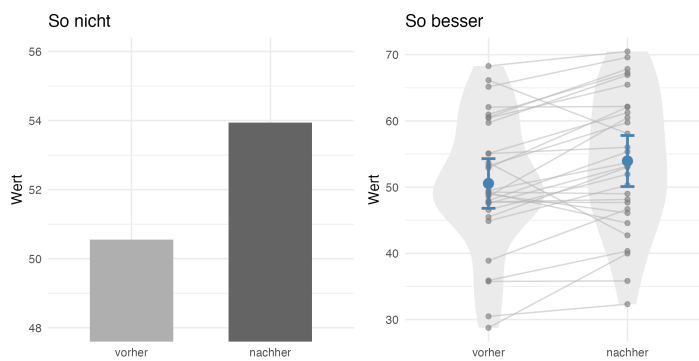


Abbildung 2.3: Derselbe Vorher-Nachher-Vergleich ($n = 30$), zweimal dargestellt. Links verletzt die Grafik gleich mehrere Prinzipien; rechts zeigt sie Rohdaten, Verbindungen, Verteilung und Mittelwert mit 95%-KI.

2.2.2 - Abbildungen für Menschen

Während diese Prinzipien schon einige Anhaltspunkte liefern, ist es auch hilfreich sich ein paar Gedanken dazu zu machen, wie das menschliche visuelle System funktioniert. Unser visuelles System ist auf bestimmte Merkmale in der Umwelt spezialisiert, die es mühelos und sehr genau unterscheiden kann. Andere visuelle Merkmale können wir Menschen nur grob und manche gar nicht unterscheiden. Ein gutes Beispiel hierfür ist das Farbsehen. Rund 8 % der Männer und 0,5 % der Frauen haben eine Farbfehlsichtigkeit, die es ihnen unmöglich macht Rot und Grün zu unterscheiden. Eine Abbildung, die ihre Information allein über die Unterscheidung von Rot und Grün transportiert, ist für diese Personen nicht lesbar. Daraus folgen zwei Regeln, die wir im praktischen Teil umsetzen: Verlassen Sie sich nie auf Farbe *allein* – stützen Sie eine Farbkodierung zusätzlich durch Form, Beschriftung oder Position ab –, und verwenden Sie farbfehlsichtigkeitssichere Farbpaletten wie das später vorgestellte *viridis*-Schema.

Dass sich Wahrnehmung von Abbildungen systematisch untersuchen lässt, macht sie zum Gegenstand einer Subdisziplin der Wahrnehmungspsychologie: der sog. **grafischen Wahrnehmung** (*graphical perception*). Die Grundfrage dieser Forschungsrichtung ist eigentlich recht einfach: Wenn eine Zahl auf eine visuelle Eigenschaft abgebildet wird – auf eine Länge, einen Winkel, eine Fläche, einen Farbton –, wie genau kann eine Betrachterin diese Zahl anschließend wieder *ablesen*? Die ersten die dies systematisch experimentell untersuchten waren Cleveland und McGill (1984). Sie erstellten unterschiedliche Abbildungen wie die in Abbildung 2.4 dargestellten und baten dann Probanden die Werte aus den einzelnen Abbildungen abzulesen. Aus der Genauigkeit mit der den Probanden dies gelang, bildeten sie eine Rangreihe von der genauesten zur ungenauesten Kodierung:

1. Position auf einer gemeinsamen Skala (z. B. Balkenenden oder Punkte an derselben Achse)
2. Position auf getrennten, aber gleich skalierten Achsen
3. Länge, Richtung, Winkel/Steigung
4. Fläche
5. Volumen, Krümmung
6. Farbsättigung und Helligkeit

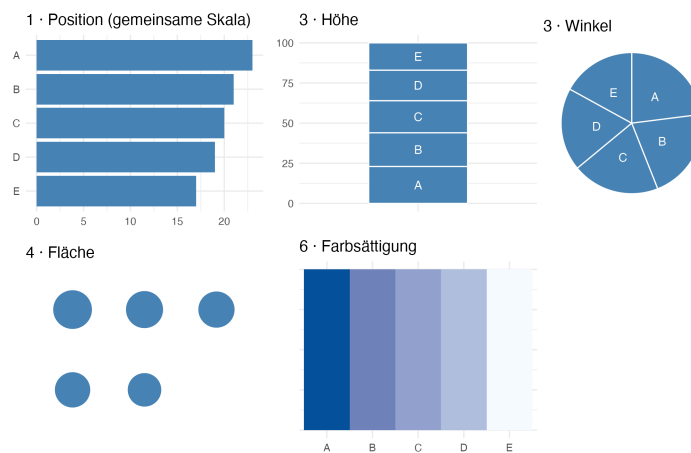


Abbildung 2.4: Dieselben fünf Werte (A–E), kodiert über verschiedene visuelle Kanäle. Mit steigender Rangziffer aus Cleveland & McGills Liste wächst die Mühe, die Werte exakt zu ordnen.

Daraus folgt eine einfache Entwurfsregel, die wir bei der praktischen Umsetzung immer wieder anwenden werden: Kodieren Sie die *wichtigste* quantitative Aussage über Position oder Länge und reservieren Sie schwächere Kanäle wie Farbe oder Fläche für kategoriale oder nachrangige Information. Wenn Sie Farben verwenden, achten Sie bitte auch darauf Farbpaletten zu verwenden, die auch für Menschen mit rot-grün Fehlsichtigkeit und auch in SW ausgedruckt zu lesen sind.

2.2.3 - Abbildungen und Unsicherheit

Bisher ging es um die *Form* der Darstellung. Ein weiterer Strang der Methodendiskussion betrifft ihren *Inhalt*: Was genau soll eine Ergebnisabbildung zeigen? Die *New Statistics* (G. Cumming & Calin-Jageman, 2017) plädieren für einen Wechsel von der dichotomen Signifikanzentscheidung („signifikant – ja oder nein?“) hin zum **Schätzen**: Im Zentrum stehen Effektgrößen und ihre Unsicherheit. Das hat auch Auswirkungen auf die Gestaltung von Abbildungen. Eine gute Ergebnisabbildung zeigt idealerweise dreierlei zugleich: die **Schätzung** (z. B. den Gruppenmittelwert), ihre **Unsicherheit** (meist ein 95%-Konfidenzintervall) und, wo möglich, die **Rohdaten**. Das ergänzt Tuftes „show the data“ um ein „show the uncertainty“.

Fehlerbalken sind dabei allerdings nur so nützlich wie ihre Beschriftung. Dieselben „Whisker“, können die **Standardabweichung** (die Streuung der Daten), den **Standardfehler** (die Präzision des Mittelwerts, etwa $SD/\sqrt{n}$) oder ein **Konfidenzintervall** (bei großem n ungefähr zwei Standardfehler) darstellen. Alle drei sehen identisch aus, bedeuten aber völlig Verschiedenes. Ein Fehlerbalken ohne Angabe, wofür er steht, ist uninterpretierbar – geben Sie es daher stets in der Bildunterschrift oder Legende an.

Cumming und Finch (2005) zeigen zudem, dass sich aus Konfidenzintervallen „mit dem Auge“, auf Signifikanz schließen lässt: Überlappen die 95%-Intervalle zweier Mittelwerte nur wenig, ist der Unterschied etwa auf dem 5%-Niveau signifikant. Abbildung 2.5 veranschaulicht

diese *rules of eye* an drei Szenarien mit unabhängigen Gruppen².

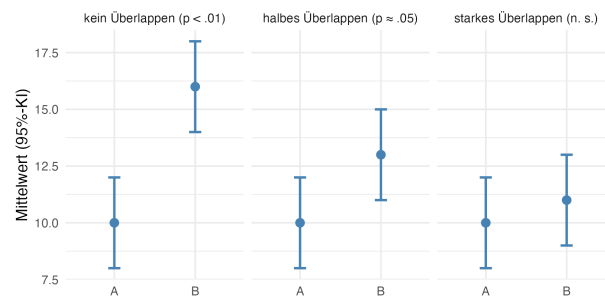


Abbildung 2.5: Rules of eye für unabhängige Gruppen: Je stärker sich zwei 95%-Konfidenzintervalle überlappen, desto größer der p-Wert. Überlappen sie sich um etwa die Hälfte einer Balkenhälfte, liegt der Unterschied nahe $p = .05$.

2.2.4 - Geklickt oder programmiert?

Grundsätzlich gibt es zwei Wege, eine Abbildung am Computer zu erzeugen. Beim ersten klicken Sie sich in einer grafischen Oberfläche (Excel, SPSS-Menüs, GraphPad) durch Dialoge und schieben Elemente von Hand zurecht. Beim zweiten *beschreiben Sie die Abbildung in Code* und lassen sie vom Programm erzeugen.

Für die tägliche Arbeit erscheint der geklickte Weg zunächst schneller. Für wissenschaftliches oder auch effizientes Arbeiten in der Wirtschaft ist der programmierte Weg jedoch klar überlegen:

- **Reproduzierbarkeit:** Der Code dokumentiert exakt und vollständig, wie die Abbildung entstanden ist. Jede*r kann sie aus den Rohdaten identisch reproduzieren. Auch Sie selbst können kritischen Kollegen ganz genau nachweisen, wie sie etwas erstellt haben.
- **Aktualisierbarkeit:** Kommen neue Daten hinzu oder ändert sich eine Kodierung, erzeugen Sie die gesamte Abbildung mit einem Tastendruck neu, ohne erneutes Zurechtklicken.
- **Konsistenz:** Über ein gemeinsames Theme sehen alle Abbildungen einer Arbeit einheitlich aus. Gerade für wiederkehrende Aufgaben wie Quartalsberichte ist das ein klarer Vorteil.
- **Skalierbarkeit:** Zwanzig Abbildungen für zwanzig Teilstichproben sind eine Schleife, keine zwanzig Nachmittage. Insofern erlauben diese Abbildungen einen wirklich ganz neuen Zugang zu den zugrundeliegenden Daten.

Diese Vorteile sind der Kern des Prinzips der **reproduzierbaren Forschung**, und sie sind der Grund, warum wir Abbildungen in R programmieren.

2.2.5 - Beispieldatensatz

Wir nutzen im weiteren Verlauf den `penguins` Datensatz aus dem Paket `palmerpenguins` als Beispiel. Er enthält Körpermaße von 344 Pinguinen dreier Arten, die auf dem Palmer-Archipel in der Antarktis erfasst wurden. Für uns interessant sind vor allem `flipper_length_mm` (Flossenlänge in mm), `body_mass_g` (Körpergewicht in Gramm) und



²Bei Messwiederholungen ist das Konfidenzintervall der *Differenz* die maßgebliche Darstellung.

`bill_length_mm` (Schnabellänge in mm) sowie die kategorialen Variablen `species` (Art) und `sex` (Geschlecht).

```
glimpse(penguins)
```

```
Rows: 344
Columns: 8
$ species      <fct> Adelie, Adelie, Adelie, Adelie,
Adelie, Adelie, Adel...
$ island       <fct> Torgersen, Torgersen, Torgersen,
Torgersen, Torgerse...
$ bill_length_mm <dbl> 39.1, 39.5, 40.3, NA, 36.7,
39.3, 38.9, 39.2, 34.1, ...
$ bill_depth_mm <dbl> 18.7, 17.4, 18.0, NA, 19.3,
20.6, 17.8, 19.6, 18.1, ...
$ flipper_length_mm <int> 181, 186, 195, NA, 193, 190,
181, 195, 193, 190, 186...
$ body_mass_g   <int> 3750, 3800, 3250, NA, 3450,
3650, 3625, 4675, 3475, ...
$ sex          <fct> male, female, female, NA, female,
male, female, male...
$ year         <int> 2007, 2007, 2007, 2007, 2007,
2007, 2007, 2007, 2007...
```

i Hinweis

Der Datensatz steckt im Paket `palmerpenguins`, das Sie ggf. einmalig mit `install.packages("palmerpenguins")` installieren müssen.

2.2.6 - Base R oder ggplot2?

R bringt bereits ein eigenes Grafiksystem mit (die Funktionen `hist()`, `boxplot()`, `plot()`), mit denen man schnell Histogramme, Streudiagramme und Boxplots erstellen kann.

```
par(mfrow = c(1, 3)) # Um Abbildungen nebeneinander
darzustellen.
hist(penguins$body_mass_g)
boxplot(penguins$body_mass_g)
plot(penguins$body_mass_g, penguins$flipper_length_mm)
```

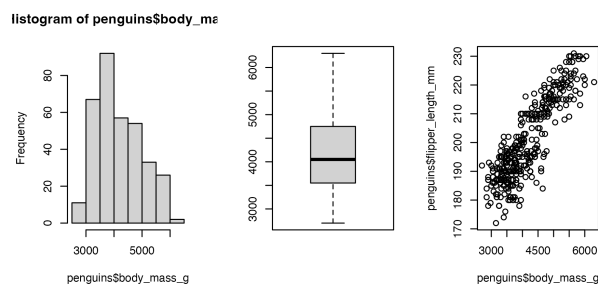


Abbildung 2.6: Einfache Histogramme, Boxplots und Streudiagramme mit dem base-R-Grafiksystem.

Für explorative Ad-hoc-Plots sind diese Arten der Abbildung völlig ausreichend. Sobald Abbildungen aber komplexer und publikationsreif werden sollen, stößt das base-R-System an Grenzen: Jede zusätzliche Ebene, jede Gruppierung, jede Legende muss von Hand ergänzt werden. Genau das wird in `ggplot2` anders gelöst und zwar auf Basis einer durchdachten Theorie von Abbildungen, der *Grammar of Graphics*.

2.3 - Die Grammar of Graphics und ggplot2

Die *Grammar of Graphics* wurde von Leland Wilkinson (2012) entwickelt und von Hadley Wickham (2010) in eine praktisch umsetzbare, „geschichtete“, (engl. layered) Form gebracht, die zur Grundlage von `ggplot2` wurde. Die Grundidee ist so elegant wie folgenreich: Statt für jeden Diagrammtyp (Balken, Linie, Streuung ...) eine eigene Spezialfunktion vorzuhalten, zerlegt man *jede* statistische Abbildung in dieselben wenigen, kombinierbaren Bausteine. Ein Diagramm ist dann keine feste Vorlage mehr, sondern eine *Grammatik*, d.h. ein Satz von Komponenten, aus denen man Abbildungen frei zusammensetzen kann. Die wichtigsten Komponenten sind in Tabelle 2.2 zusammengefasst.



Tabelle 2.2: Bereits die ersten drei sind ausreichend um lesbare Abbildungen etwa auf demselben Level wie die `hist` zu erstellen. Allerdings ermöglichen gerade die weiteren Komponenten deutlich bessere Abbildungen.

Komponente	Bedeutung	In <code>ggplot2</code>
Data	die zugrunde liegenden Daten	<code>ggplot(data = ...)</code>
Aesthetics	Zuordnung von Variablen zu grafischen Eigenschaften (Achsen, Farbe, Größe ...)	<code>aes(...)</code>
Geometries	die geometrische Repräsentation (Punkte, Linien, Balken ...)	<code>geom_*()</code>
Statistics	statistische Transformationen der Daten (Zählen, Glätten ...)	<code>stat_*()</code>
Scales	Übersetzung von Datenwerten in konkrete Achsen- und Farbskalen	<code>scale_*()</code>
Facets	Aufteilung in Teilabbildungen (Small Multiples)	<code>facet_*()</code>
Coordinates	das Koordinatensystem	<code>coord_*()</code>
Theme	das nicht-datenbezogene Erscheinungsbild	<code>theme_*()</code>

Der entscheidende Punkt: Diese Bausteine sind unabhängig voneinander und lassen sich frei kombinieren. Wechseln Sie das `geom` von Punkten zu Linien, bleibt alles andere gleich. Fügen Sie eine Farb-Aesthetic hinzu, gilt sie automatisch für alle Ebenen. Aus wenigen Bausteinen entsteht so eine praktisch unbegrenzte Vielfalt von Abbildungen.

2.3.1 - Der Grundaufbau einer `ggplot`-Abbildung

Jede `ggplot2`-Abbildung beginnt mit der Funktion `ggplot()`, der Sie die Daten und über `aes()` die *Aesthetic Mappings* übergeben. Für sich genommen erzeugt das nur eine leere Zeichenfläche mit passend skalierten Achsen, es fehlt noch das `geom`, das festlegt, wie die Daten dargestellt werden.

```
ggplot(data = penguins, mapping = aes(x = flipper_length_mm, y = body_mass_g))
```

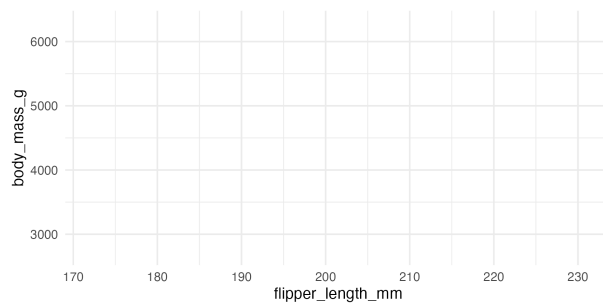


Abbildung 2.7: Nur `ggplot()` mit `aes()`: Die Achsen sind angelegt, aber es wird noch nichts gezeichnet.

Ebenen aka Layers werden mit dem `+`-Operator hinzugefügt. Fügen wir `geom_point()` hinzu, entsteht ein Streudiagramm:

```
ggplot(penguins, aes(x = flipper_length_mm, y =  
body_mass_g)) +  
  geom_point()
```

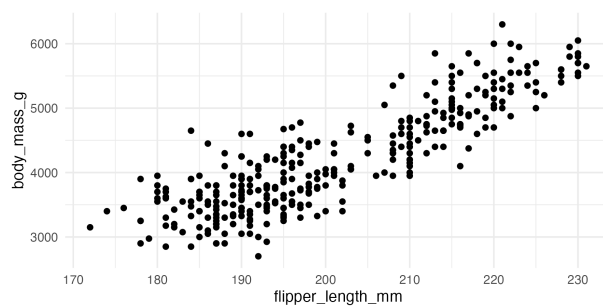


Abbildung 2.8: Dieselbe Abbildung mit `geom_point()`: ein Streudiagramm.

Man erkennt sofort einen klaren positiven Zusammenhang: Pinguine mit längeren Flossen sind in der Regel schwerer. Übrigens sind die Argumentnamen `data =` und `mapping =` optional, da sie an erster und zweiter Stelle stehen, lässt man sie üblicherweise weg, wie in diesem und allen folgenden Beispielen.

2.3.2 - Aesthetics: Variablen abbilden

Eine *Aesthetic* ist eine visuelle Eigenschaft, auf die eine Variable *abgebildet* wird. Neben `x` und `y` gehören dazu insbesondere `color`, `fill`, `shape`, `size` und `alpha` (Transparenz). Bilden wir die Art auf die Farbe ab:

```
ggplot(penguins, aes(x = flipper_length_mm, y =  
body_mass_g, color = species)) +  
  geom_point()
```

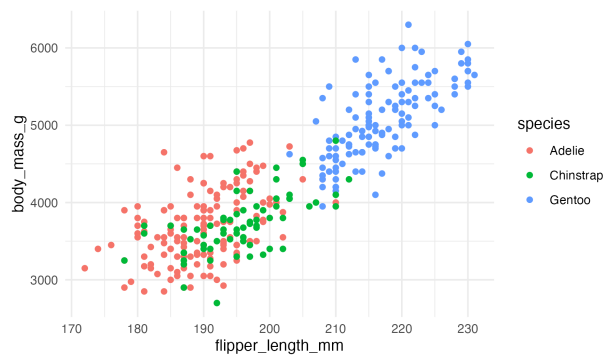


Abbildung 2.9: Die Variable `species` wird auf die Aesthetic color abgebildet; `ggplot2` ergänzt automatisch eine passende Legende.

Nun wird deutlich, dass sich die Punktwolke in die drei Arten auffächert: Die Gentoo-Pinguine bilden die Gruppe mit langen Flossen und hohem Gewicht oben rechts, während sich Adelia und Chinstrap im unteren Bereich stärker überlappen.

2.3.3 - Geome

Das `geom` bestimmt die geometrische Repräsentation. Dieselben Daten lassen sich durch einfaches Austauschen des `geom` völlig unterschiedlich darstellen. Ein Boxplot des Körpergewichts pro Art:

```
ggplot(penguins, aes(x = species, y = body_mass_g)) +  
  geom_boxplot()
```

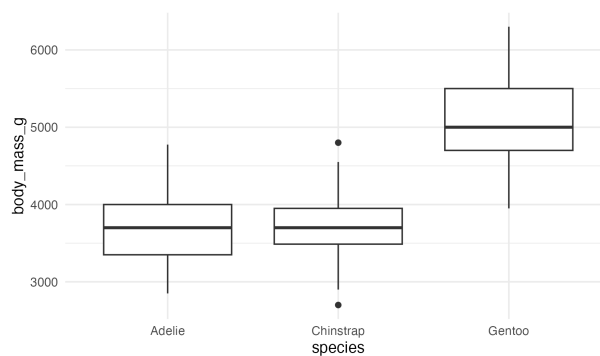


Abbildung 2.10: Körpergewicht nach Art als Boxplot.

Für eine einzelne kontinuierliche Variable eignen sich Histogramm oder Dichtekurve:

```
ggplot(penguins, aes(x = flipper_length_mm)) +  
  geom_histogram(binwidth = 2)
```

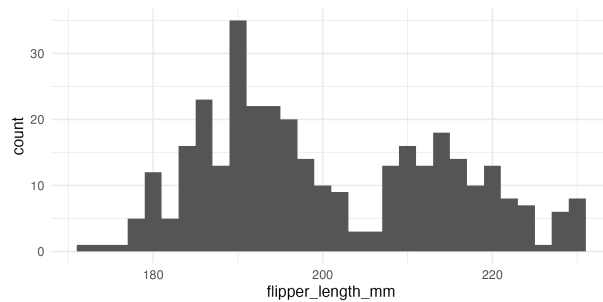


Abbildung 2.11: Verteilung der Flossenlänge als Histogramm.

Die deutliche Zweigipfligkeit der Flossenlänge ist ein typischer EDA-Befund: Sie deutet auf Untergruppen hin – tatsächlich hat die (größere) Gentoo-Art erheblich längere Flossen als die beiden anderen Arten. In keinem einzelnen Kennwert wäre diese Struktur sichtbar gewesen.

i Hinweis

Beim Histogramm lohnt es sich fast immer, die Klassenbreite bewusst über `binwidth` (oder die Anzahl über `bins`) zu setzen. Der Standardwert (`bins = 30`) ist selten optimal, und `ggplot2` weist Sie mit einer Meldung darauf hin.

2.3.4 - Mehrere Ebenen kombinieren

Die volle Stärke der Grammatik zeigt sich, wenn Sie mehrere `geoms` übereinanderlegen. Wir kombinieren die Rohdaten (Punkte) mit einer geglätteten Trendlinie:

```
ggplot(penguins, aes(x = flipper_length_mm, y =
  body_mass_g)) +
  geom_point() +
  geom_smooth(method = "lm")
```

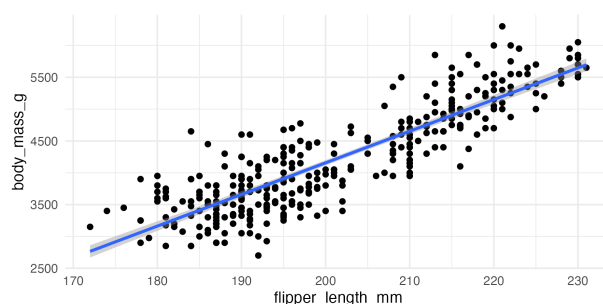


Abbildung 2.12: Zwei Ebenen: Rohdaten als Punkte, überlagert von einer linearen Trendlinie mit Konfidenzband.

Aesthetics, die Sie im `ggplot()`-Aufruf angeben, gelten für *alle* Ebenen. Ergänzen wir eine Farb-Aesthetic für die Art, wird sowohl nach Farbe gepunktet als auch je Gruppe eine eigene Gerade geschätzt:

```
ggplot(penguins, aes(x = flipper_length_mm, y =
body_mass_g, color = species)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

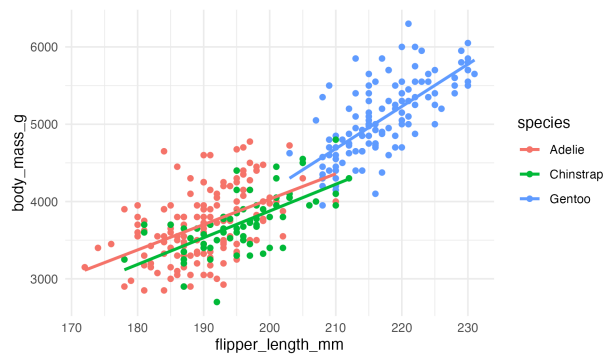


Abbildung 2.13: Eine globale color-Aesthetic wirkt auf beide Ebenen: getrennte Punkte und getrennte Trendgeraden je Art.

2.3.5 - Statistische Transformationen

Manche `geoms` zeichnen nicht die Rohdaten, sondern eine *statistische Transformation* davon. `geom_bar()` etwa zählt im Hintergrund die Häufigkeiten je Kategorie (die Statistik `count`):

```
ggplot(penguins, aes(x = species)) +
  geom_bar()
```

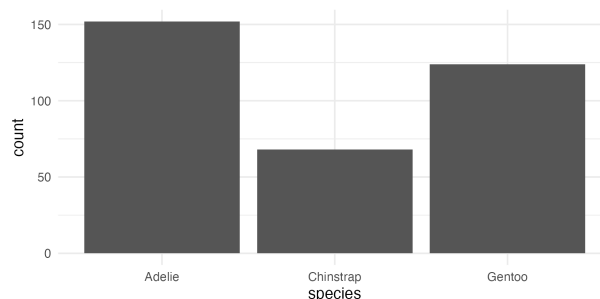


Abbildung 2.14: `geom_bar()` zählt automatisch die Fälle je Art.

Häufig möchten Sie aber nicht Häufigkeiten, sondern *Zusammenfassungen* wie Mittelwerte mit Streuungsmaß darstellen. `stat_summary()` erlaubt das flexibel:

```
ggplot(penguins, aes(x = species, y = body_mass_g)) +
  stat_summary(fun = mean, geom = "point", size = 3) +
  stat_summary(fun.data = mean_se, geom = "errorbar",
width = 0.2)
```

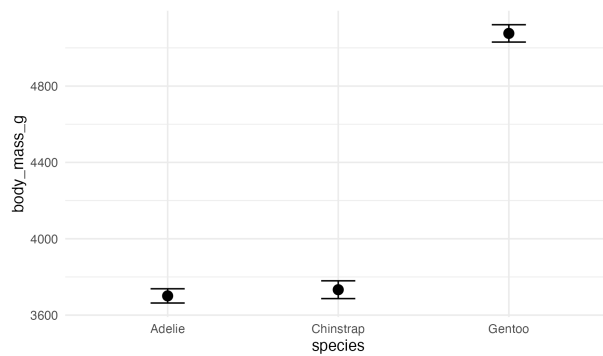


Abbildung 2.15: Mittelwert und Standardfehler des Körpergewichts je Art, direkt aus den Rohdaten berechnet.

Den größten Erkenntniswert hat eine solche Zusammenfassung, wenn sie die Rohdaten *und* die Unsicherheit der Schätzung zeigt (siehe Abschnitt „Abbildungen und Unsicherheit“). Für ein 95%-Konfidenzintervall berechnen wir dessen Grenzen einmalig selbst und legen Mittelwert und Fehlerbalken über die leicht gejitterten Rohdaten:

```
ggplot(penguins, aes(x = species, y = body_mass_g)) +  
  geom_jitter(width = 0.08, alpha = 0.3) +  
  stat_summary(fun.data = mean_cl_boot, col = "red") +  
  labs(x = "Art", y = "Körpergewicht (g)")
```

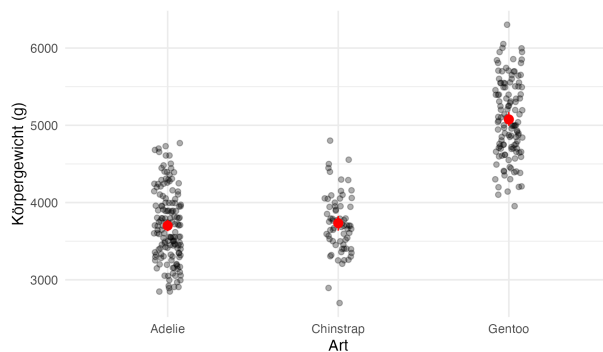


Abbildung 2.16: Estimation-Plot: Rohdaten (grau), Gruppenmittelwert und 95%-Konfidenzintervall des Mittelwerts je Art.

💡 Tipp

Die Bildunterschrift nennt ausdrücklich, wofür die Fehlerbalken stehen (95%-KI des Mittelwerts). Ohne diese Angabe wäre die Abbildung nicht interpretierbar, da unklar wäre, ob die Fehlerbalken Konfidenzintervalle, eine Standardabweichung oder einen Standardfehler zeigen.

2.3.6 - Position adjustments

Wenn sich Elemente überlagern, steuert das `position`-Argument, wie sie angeordnet werden. Bei gruppierten Balken sind die drei wichtigsten Varianten `"stack"`, `"dodge"` und `"fill"` (vgl. Abb. 2.17). Der Standard `"stack"` stapelt die Gruppen aufeinander. `"dodge"` stellt die Balken nebeneinander — am besten geeignet, um die Gruppen direkt zu vergleichen. Zuletzt normiert `"fill"` jede Säule auf 100 % und zeigt damit Anteile statt absoluter Häufigkeiten:

```
library(patchwork)

penguin_sex <- drop_na(penguins, sex)

p_stack <- ggplot(penguin_sex, aes(x = species, fill = sex)) +
  geom_bar(position = "stack") +
  labs(title = "stack", x = NULL, y = "Anzahl")

p_dodge <- ggplot(penguin_sex, aes(x = species, fill = sex)) +
  geom_bar(position = "dodge") +
  labs(title = "dodge", x = NULL, y = "Anzahl")

p_fill <- ggplot(penguin_sex, aes(x = species, fill = sex)) +
  geom_bar(position = "fill") +
  labs(title = "fill", x = NULL, y = "Anteil")

p_stack + p_dodge + p_fill + plot_layout(guides = "collect")
```

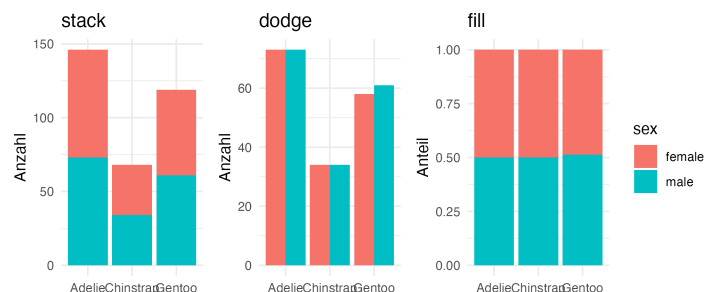


Abbildung 2.17: Dieselben gruppierten Daten mit drei Positionsanpassungen: gestapelt (stack), nebeneinander (dodge) und auf 100 % normiert (fill).

Für Streudiagramme mit vielen überlappenden Punkten ist `position = "jitter"` nützlich: Es verrauscht die Positionen minimal und macht so die tatsächliche Punktdichte sichtbar. Die Kurzform ist `geom_jitter()`.

2.3.7 - Skalen

Scales übersetzen Datenwerte in konkrete visuelle Werte — welche Zahl liegt wo auf der Achse, welche Kategorie bekommt welche Farbe. Für jede Aesthetic gibt es passende `scale_*()`-Funktionen.

Besonders häufig passt man **Farbskalen** an. Hier kommt die im konzeptuellen Teil angesprochene Barrierefreiheit ins Spiel: Die Standardpalette von `ggplot2` ist für farbfehlsichtige Personen nicht ideal. Die *viridis*-Paletten wurden genau dafür entworfen. Sie sind *perzeptuell uniform* – gleiche Abstände im Farbwert entsprechen gleichen wahrgenommenen Abständen –, bleiben auch bei einer Rot-Grün-Schwäche unterscheidbar und funktionieren sogar in Graustufen. Sie sind direkt in `ggplot2` eingebaut (kein Zusatzpaket nötig) und über `scale_color_viridis_*`() bzw. `scale_fill_viridis_*`() erreichbar. Das Suffix wählt den Skalentyp: `_d` für diskrete (kategoriale), `_c` für kontinuierliche Variablen.

```
ggplot(penguins, aes(x = flipper_length_mm, y =
  body_mass_g, color = species)) +
  geom_point() +
  scale_color_viridis_d()
```

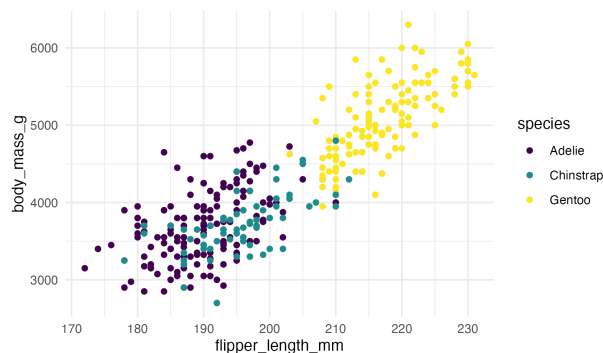


Abbildung 2.18: Diskrete viridis-Farbskala (`scale_color_viridis_d`) statt der Standardfarben.

Auch **Achsen** sind Skalen. Über `scale_*_continuous()` steuern Sie die Achsenabschnitte (`breaks`), deren Beschriftung (`labels`) und den sichtbaren Bereich (`limits`):

```
ggplot(penguins, aes(x = flipper_length_mm, y =
  body_mass_g)) +
  geom_point(alpha = 0.6) +
  scale_y_continuous(
    breaks = seq(3000, 6000, by = 1000),
    labels = scales::label_number(suffix = " g")
  )
```

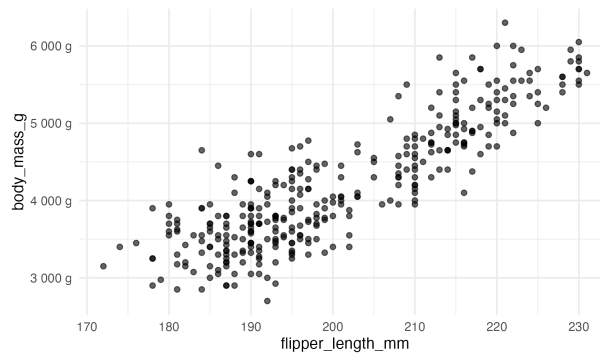


Abbildung 2.19: Angepasste y-Achse: eigene Achsenabschnitte und eine Beschriftung mit Einheit.

Für stark rechtsschiefe Variablen — etwa Einkommen oder Reaktionszeiten — bietet sich zudem eine logarithmische Achse an, die Sie einfach über `scale_x_log10()` bzw. `scale_y_log10()` erhalten.

2.3.8 - Facetten: Small Multiples

Das *Facetting* setzt das oben beschriebene Small-Multiples-Prinzip um: Es zerlegt die Daten nach einer (oder zwei) kategorialen Variablen und zeichnet dieselbe Abbildung für jede Teilmenge. `facet_wrap()` teilt nach einer Variablen auf:

```
ggplot(penguins, aes(x = flipper_length_mm, y = body_mass_g)) +
  geom_point() +
  facet_wrap(~ species)
```

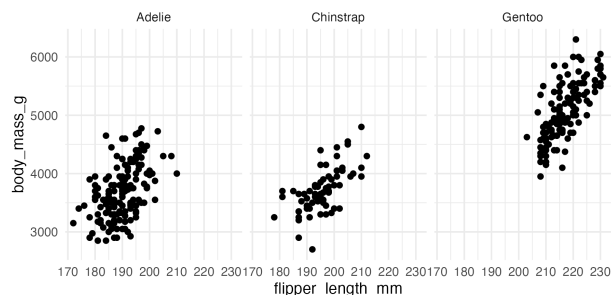


Abbildung 2.20: Der Flossenlänge-Gewicht-Zusammenhang, getrennt für jede Art (`facet_wrap`).

`facet_grid()` bildet ein Gitter aus zwei Variablen (Zeilen ~ Spalten) und eignet sich, um zwei Faktoren gleichzeitig zu betrachten:

```
ggplot(drop_na(penguins, sex), aes(x = flipper_length_mm, y = body_mass_g)) +
  geom_point() +
  facet_grid(species ~ sex)
```

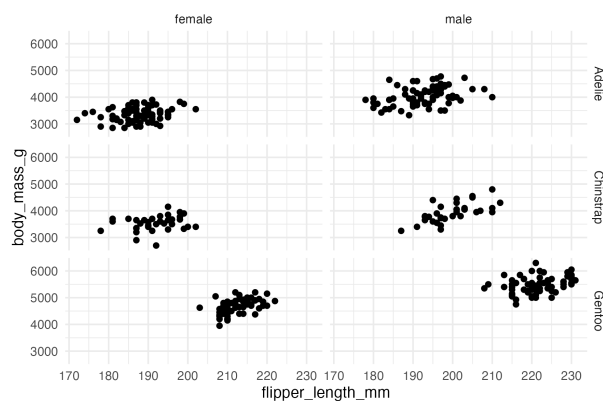


Abbildung 2.21: Ein Facetten-Gitter aus Art (Zeilen) und Geschlecht (Spalten).

Facetting ist eines der wirksamsten Mittel überhaupt, um komplexe, mehrdimensionale Zusammenhänge übersichtlich darzustellen – und in `ggplot2` kostet es genau eine Zeile.

2.3.9 - Koordinatensysteme

Standardmäßig verwendet `ggplot2` ein kartesisches Koordinatensystem. `coord_flip()` vertauscht die Achsen, was bei langen Kategorienamen die Lesbarkeit verbessert. `coord_cartesian()` erlaubt es, in einen Ausschnitt zu „zoomen“, ohne dabei Datenpunkte zu entfernen (was Kennwerte wie Trendlinien verfälschen würde):

```
ggplot(penguins, aes(x = species, y = body_mass_g)) +
  geom_boxplot() +
  coord_flip()
```

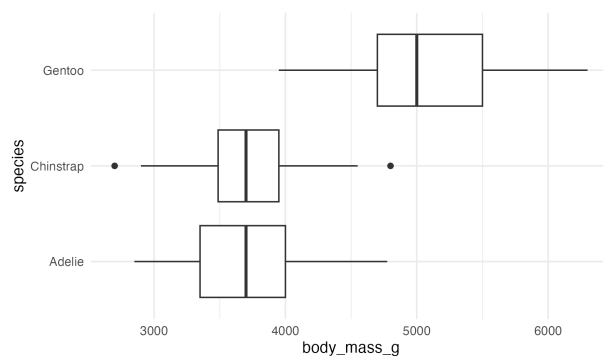


Abbildung 2.22: `coord_flip()` dreht den Boxplot; die Arten stehen nun an der vertikalen Achse.

2.3.10 - Beschriftung und Themes

Zu einer publikationsreifen Abbildung gehören informative Beschriftungen. `labs()` steuert Titel, Achsen- und Legendentitel sowie eine Quellenangabe (`caption`). Das *Theme* bestimmt das gesamte nicht-datenbezogene Erscheinungsbild – Schriftgrößen, Gitterlinien, Hintergrund. `ggplot2` bringt eine Reihe fertiger Themes mit

(`theme_minimal()`, `theme_classic()`, `theme_bw()` ...), und einzelne Elemente lassen sich über `theme()` gezielt anpassen:

```
ggplot(penguins, aes(x = flipper_length_mm, y =
  body_mass_g, color = species)) +
  geom_point() +
  scale_color_viridis_d() +
  labs(
    title = "Längere Flossen, schwerere Pinguine",
    subtitle = "Körpermaße von 344 Pinguinen des Palmer-
Archipels",
    x = "Flossenlänge (mm)",
    y = "Körpergewicht (g)",
    color = "Art",
    caption = "Datenquelle: Palmer Station LTER /
palmerpenguins"
  ) +
  theme_classic(base_size = 11)
```

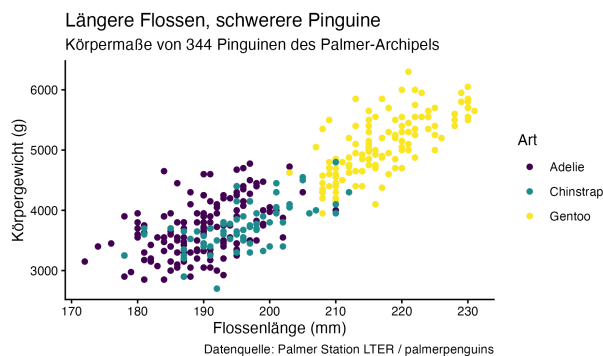


Abbildung 2.23: Vollständig beschriftete Abbildung mit angepasstem Theme (`theme_classic`) und viridis-Farbskala.

2.3.11 - Abbildungen speichern

In einem Quarto-Dokument werden Abbildungen beim Rendern automatisch eingebunden. Möchten Sie eine Abbildung als eigenständige Datei exportieren — etwa für einen Vortrag oder eine Zeitschrift —, verwenden Sie `ggsave()`. Es speichert standardmäßig die zuletzt erzeugte Abbildung; Format, Größe und Auflösung geben Sie explizit an:

```
ggsave("pinguine.png", width = 6, height = 4, dpi = 300)
```

Das Dateiformat ergibt sich aus der Endung. Für Publikationen empfiehlt sich ein Vektorformat wie `.pdf` oder `.svg`, das sich verlustfrei skalieren lässt.

2.3.12 - Ein zusammenfassendes Beispiel

Zum Abschluss kombinieren wir die Bausteine zu einer einzigen, kompakten Abbildung. Sie verknüpft zwei kontinuierliche Variablen (x , y), eine Farb-Aesthetic (`species`), zwei geometrische Ebenen (Punkte und Trendlinien), Facetting nach Erhebungsjahr, eine angepasste Farbskala, vollständige Beschriftung und ein reduziertes Theme:

```
ggplot(penguins, aes(x = flipper_length_mm, y =
body_mass_g, color = species)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ year) +
  scale_color_viridis_d(option = "C", end = 0.85) +
  labs(
    title = "Flossenlänge und Körpergewicht nach Art
und Erhebungsjahr",
    x = "Flossenlänge (mm)",
    y = "Körpergewicht (g)",
    color = "Art"
  ) +
  theme_minimal(base_size = 11)
```

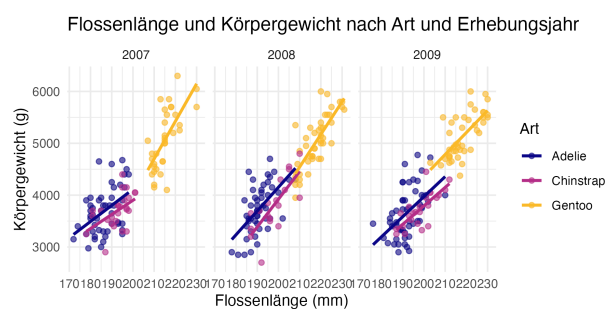


Abbildung 2.24: Eine Abbildung, viele Bausteine der Grammar of Graphics: Aesthetics, zwei Geome, Facetten, eine angepasste Skala, Beschriftung und Theme.

Jeder einzelne Baustein ist für sich verständlich, und zusammen ergeben sie eine dichte, ehrliche und reproduzierbare Darstellung. Genau das ist die Idee der Grammar of Graphics.

2.3.13 - Mehrere Abbildungen kombinieren: cowplot

Manchmal soll eine einzige Abbildung mehrere eigenständige Diagramme vereinen – etwa eine mehrteilige „Panel“-Abbildung mit den Teilbeschriftungen A, B, C, wie sie in Publikationen üblich ist. Das im konzeptuellen Teil bereits genutzte *patchwork* ist eine Möglichkeit; eine zweite, gerade für Publikationsabbildungen sehr verbreitete ist das Paket *cowplot*. Seine zentrale Funktion ist `plot_grid()`: Sie ordnet beliebige ggplots in einem Raster an, versieht sie auf Wunsch mit Teilbeschriftungen (`labels`), richtet ihre Achsen exakt aneinander aus (`align, axis`) und kann eine gemeinsame Legende teilen.

```
library(cowplot)

p_scatter <- ggplot(penguins, aes(flipper_length_mm,
body_mass_g, color = species)) +
  geom_point() +
  scale_color_viridis_d() +
  theme_minimal(base_size = 10)

p_box <- ggplot(drop_na(penguins, sex), aes(species,
```

```
body_mass_g, fill = species)) +
  geom_boxplot() +
  scale_fill_viridis_d(guide = "none") +
  theme_minimal(base_size = 10)

plot_grid(p_scatter, p_box, labels = c("A", "B"),
rel_widths = c(1.5, 1))
```

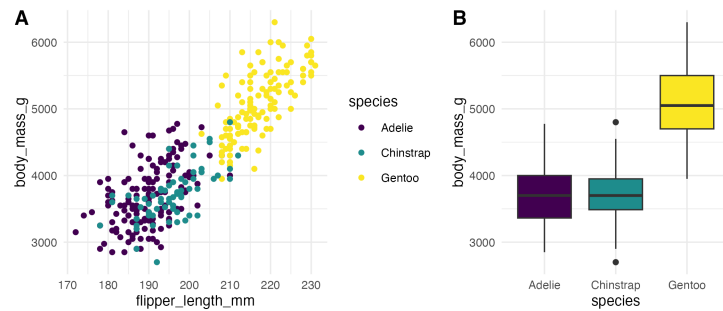


Abbildung 2.25: Zwei eigenständige Abbildungen, mit `cowplot::plot_grid()` zu einer beschrifteten Panel-Abbildung kombiniert.

i Hinweis

`cowplot` und das unten verwendete `ggupset` sind Zusatzpakete: `install.packages(c("cowplot", "ggupset"))`.

Ein aufschlussreiches Anwendungsbeispiel für zusammengesetzte Abbildungen sind **UpSet-Plots**. Sie visualisieren die Schnittmengen zwischen *Mengen*: Wenn ein Element mehreren Gruppen zugleich angehören kann (ein Film ist Komödie *und* Drama), wird ein Venn-Diagramm ab drei oder vier Mengen unübersichtlich. Ein UpSet-Plot zeigt stattdessen die Größe jeder Schnittmenge als Balken, ausgerichtet an einer Matrix, die die beteiligten Mengen markiert – selbst also eine aus mehreren exakt ausgerichteten Teilabbildungen zusammengesetzte Grafik. Das Paket `ggupset` erzeugt sie als gewöhnliche ggplots über `scale_x_upset()`:

```
library(ggupset)

tidy_movies |>
  distinct(title, year, length, .keep_all = TRUE) |>
  ggplot(aes(x = Genres)) +
  geom_bar() +
  scale_x_upset(n_intersections = 8) +
  labs(x = "Genre-Kombination", y = "Anzahl Filme")
```

cowplot

ggupset

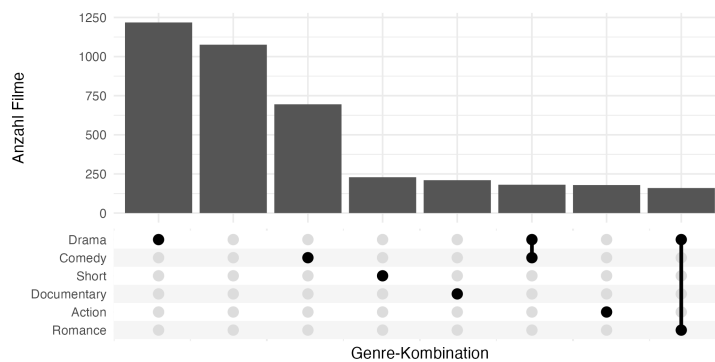


Abbildung 2.26: UpSet-Plot der häufigsten Genre-Kombinationen von Filmen (ggupset). Jede Säule steht für eine Kombination, die Matrix darunter nennt die beteiligten Genres.

Da `ggupset` ein gewöhnliches `ggplot`-Objekt zurückgibt, lässt es sich mit `plot_grid()` oder `patchwork` problemlos neben weitere Panels setzen — damit schließt sich der Kreis zu `cowplot`.

2.4 - Fazit

- Abbildungen dienen zwei Zwecken: der **Prüfung und Exploration** der eigenen Daten und der **Kommunikation** von Ergebnissen. Anscombes Quartett zeigt, warum der erste Zweck unverzichtbar ist: Kennwerte allein können täuschen.
- Gute Abbildungen folgen wenigen Prinzipien: grafische Integrität, ein hoher Data-Ink-Anteil, hohe Datendichte und die Sichtbarkeit der Rohdaten.
- Wissenschaftliche Abbildungen sollten **programmiert** statt geklickt werden — der Reproduzierbarkeit, Aktualisierbarkeit und Konsistenz wegen.
- Die **Grammar of Graphics** zerlegt jede Abbildung in kombinierbare Bausteine: Daten, Aesthetics, Geome, Statistiken, Skalen, Facetten, Koordinaten und Theme.
- `ggplot2` setzt diese Grammatik in R um. Wenn Sie die Bausteine beherrschen, erstellen Sie aus ihnen praktisch jede Abbildung, die Sie brauchen.

2.4.1 - Vertiefung

Wenn Sie an dem Thema Abbildungen interessiert sind und sich auf einer etwas abstrakteren Ebene mit Abbildungen beschäftigen möchten, empfehle ich Ihnen unbedingt die Bücher von Edward Tufte. Es gibt auch eine sehr gute Keynote von Ihm, die er vor ein paar Jahren auf einem Microsoft summit gehalten hat (<https://www.youtube.com/watch?v=rHUDJ8RyseQ>). Ein eher praktisches und weniger tiefgehendes Buch zum Thema Datenvisualisierung ist das Buch von Alberto Cairo (2012). Falls Sie besonders viel über `ggplot2` lernen möchten, empfehle ich das komplette Buch über das `ggplot2`-Paket von dem Entwickler selbst (Wickham, 2016), sowie das frei verfügbare Kapitel zur Datenvisualisierung in [R for Data Science](#).

3. Datenmanagement

„If you torture the data long enough, it will confess to anything.“
– zugeschrieben R. Coase

In diesem Kapitel lernen Sie, warum ein sorgfältiges und vor allem *nachvollziehbares* Datenmanagement über die Glaubwürdigkeit einer ganzen Studie entscheiden kann, welche konzeptuellen Prinzipien der Aufbereitung von Daten zugrunde liegen und wie Sie mit dem `tidyverse` sehr effizient von den Rohdaten zu einem analysefertigen Datensatz kommen. Wie im vorigen Kapitel ist die erste Hälfte bewusst konzeptuell: Sie soll Ihnen ein Gespür dafür geben, *warum* wir Daten so und nicht anders behandeln. Die zweite Hälfte ist praktisch und zeigt Schritt für Schritt, wie Sie in R aus einer Roh-Tabelle einen sauberen, dokumentierten Datensatz machen, mit dem man gut arbeiten kann.

Wie sehr die Weltwirtschaft an einer Excel-Formel hängen kann, zeigt eine der bekanntesten Episoden der jüngeren Wissenschaftsgeschichte. Die Ökonomen Carmen Reinhart und Kenneth Rogoff (2010) veröffentlichten 2010 ihre Studie *Growth in a Time of Debt* mit einer alarmierenden These: Übersteige die Staatsverschuldung eines Landes **90 % des Bruttoinlandsprodukts**, breche das Wirtschaftswachstum abrupt ein. Sie hatten auch einen schmissigen Namen für diesen Effekt das „growth cliff“. Die Arbeit wurde zu einer der meistzitierten empirischen Grundlagen für die Austeritätspolitik nach der Finanzkrise von 2008.

Als der Doktorand Thomas Herndon gemeinsam mit Michael Ash und Robert Pollin (2013) versuchte, die Ergebnisse zu replizieren, bekam er zunächst schlicht andere Zahlen. Erst der Zugang zur Original-Tabelle brachte drei Probleme ans Licht: Eine **Excel-Formel** hatte fünf Länder aus der Mittelwertsberechnung ausgeschlossen, weil der markierte Zellbereich nicht bis zum Ende reichte. Hinzu kamen eine ungewöhnliche **Gewichtung** der Länder und der **selektive Ausschluss** einzelner Jahre. Rechnet man diese drei Punkte heraus, steigt das mittlere Wachstum bei einer Verschuldung über 90 % von berichteten **-0,1 %** auf **+2,2 %** und der „cliff“ verschwindet.

	B	C	I	J	K	L	M
	Real GDP growth						
	Units: 1981-1990						
Country	Coverage	30 or less	30 to 50	50 to 60	60 or more	90 or above	30 or less
26	Minimum	na.	3.7	0.0	3.3	1.7	5.3
27	Maximum	na.	6.4	0.3	1.3	-1.8	1.8
28			5.4	4.8	10.2	3.6	11.3
29							
30	UK	1946-1959	na.	3.4	3.3	-2.0	na.
31	UK	1960-1969	na.	2.4	2.5	-1.8	3.2
32	Sweden	1946-1959	3.6	2.9	2.7	na.	6.3
33	Spain	1946-1959	1.5	1.4	4.2	na.	7.9
34	France	1952-1959	4.4	2.7	1.1	na.	8.9
35	New Zealand	1948-1959	2.5	2.9	3.9	-7.9	2.6
36	Netherlands	1956-1969	4.4	2.7	1.1	na.	6.4
37	Belgium	1951-1959	5.4	5.1	na.	na.	11.3
38	Japan	1946-1959	7.0	4.0	1.0	0.7	7.0
39	Italy	1951-1959	2.7	2.1	1.8	1.0	7.6
40	Ireland	1948-1959	4.4	4.5	4.0	na.	11.3
41	Greece	1970-2009	4.0	0.3	2.7	2.9	13.3
42	Germany	1946-1959	na.	3.9	na.	na.	3.2
43	France	1949-1959	4.9	2.7	3.0	na.	5.2
44	Finland	1946-1959	3.9	2.8	2.4	5.5	na.
45	Denmark	1950-1959	2.4	2.4	2.4	na.	5.6
46	Canada	1951-1959	1.9	3.6	3.1	na.	2.2
47	Belgium	1947-1959	na.	4.2	3.3	2.6	na.
48	Australia	1948-1959	na.	1.3	3.8	-2.7	na.
49	Australia	1951-2009	3.2	4.9	4.0	na.	5.9
50							
51			2.1	3.8	2.8	0.4	na.

3.1 - Folgen für das Datenmanagement

Die wichtigste Lehre, die man aus diesem Fall ziehen kann, ist dass jeder Fehler machen kann und wir es anderen möglichst leicht machen sollten, diese zu finden. Das bedeutet einerseits, dass man die Daten möglichst einfach und frei zugänglich machen sollte. Zweitens sollte man sein Vorgehen möglichst transparent dokumentieren. Also am besten keinen einzigen Schritte von Hand durchführen, sondern alle Schritte vom Einlesen über das Auswählen, Transformieren, und Umstrukturieren bis zur Erstellung von Abbildungen und Ergebnistabellen durchgängig zu dokumentieren. Im folgenden gehe ich auf beides kurz ein.

Abbildung 3.1: Der kritische Ausschnitt aus der Original-Tabelle: Formel zur Berechnung der Mittelwerte schließt die Länder von Dänemark bis Australien einfach nicht mit ein. Quelle: <https://statmodeling.stat.columbia.edu>

3.1.1 - Daten und Code sollten nach FAIR geteilt werden

Warum sollte man Daten überhaupt mit anderen teilen? Der Fall Reinhart-Rogoff liefert das erste Argument gleich mit: **Reproduzierbarkeit**. Nur wenn andere Zugang zu den Daten haben, können sie Ergebnisse prüfen und Fehler finden. Das zweite Argument ist die **Nachnutzung**: Datenerhebung ist teuer, und viele Datensätze erlauben weit mehr als die eine Auswertung, für die sie erhoben wurden – Zweitanalysen und Meta-Analysen leben davon. Das dritte Argument ist **Vertrauen**: Offenheit wirkt als Korrektiv gegen ehrliche Fehler und, wie wir gleich sehen werden, auch gegen Betrug. Für die Veröffentlichung stehen fachübergreifende Repositorien wie OSF, Zenodo oder GESIS bereit.

Nach den FAIR Prinzipien sollen Daten **F**indable (auffindbar), **A**ccessible (zugänglich), **I**nteroperable (mit anderen Daten kombinierbar) und **R**eusable (nachnutzbar) sein (M. D. Wilkinson et al., 2016).

Um das zu erreichen, fordern mittlerweile viele Große Förderorganisationen wie die Deutsche Forschungsgemeinschaft (DFG) oder das EU-Rahmenprogramm Horizon Europe inzwischen sog. **Datenmanagementpläne** (DMP), in denen Sie schon vor der Erhebung festlegen, wie die Daten strukturiert, dokumentiert, gespeichert und später zugänglich gemacht werden. Im Prinzip entspricht das zum großen Teil schon dem Forschungsprozess in der Psychologie, nur mit einem stärkeren Fokus auf den Daten und deren Speicherung. Datenmanagementpläne helfen aber dabei den Prozess in Bezug auf die Daten zu strukturieren.

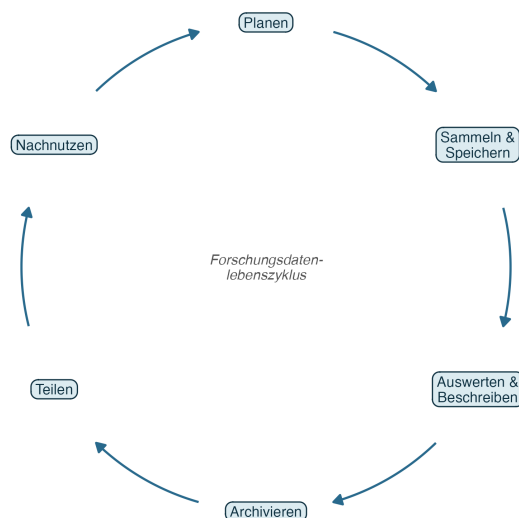


Abbildung 3.2: Der Forschungsdatenlebenszyklus als geschlossener Kreislauf (nach dem UK-Data-Service-Modell, vgl. Corti (2020)).

Ein sauberer Datensatz nützt wenig, wenn niemand weiß, was in ihm steht. Zu jeder Erhebung gehört daher eine Dokumentation, die üblicherweise aus mehreren Bausteinen besteht. Das **Codebook** (oder Datenwörterbuch) hält für jede Variable Namen, Bedeutung, zulässigen Wertebereich und die Kodierung fehlender Werte fest. Die **Metadaten** beschreiben den Entstehungskontext: wer wann womit erhoben hat. Und schließlich verlangt der **Datenschutz**, dass personenbezogene Daten vor jeder Weitergabe anonymisiert oder zumindest pseudonymisiert werden – in Europa ist das durch die Datenschutz-Grundverordnung (DSGVO) rechtlich verbindlich.

3.1.2 - Transparente Dokumentation der Datenverarbeitungsschritte

Ein anderes Wichtiges Prinzip, dass man befolgen sollte ist, dass die Rohdaten selbst **unantastbar** (read-only) sind, und jede Transformation passiert im **Skript** und nie per Hand. Der Unterschied ist fundamental. Wer sich in einer Tabellenkalkulation durch die Daten klickt, sortiert, einzelne Zeilen löscht und Werte überschreibt, gelangt vielleicht zu einem Ergebnis, kann aber niemandem und auch nicht sich selbst ein Jahr später genau erklären, wie er dorthin gekommen ist. Wer stattdessen jeden Schritt als Code notiert, erhält ein Protokoll, das die Analyse vollständig und exakt beschreibt:

- **Von Hand:** Rohdaten → klicken, sortieren, löschen → Ergebnis. *Nicht nachvollziehbar.*
- **Im Skript:** Rohdaten → dokumentierter Code → Ergebnis. *Reproduzierbar.*

3.2 - Tidy Data

„Tidy datasets are all alike, but every messy dataset is messy in its own way.“ – Hadley Wickham (2014)

Wenn jede Transformation im Skript passieren soll, brauchen die Daten eine Struktur, mit der ein Programm zuverlässig arbeiten kann. Hadley Wickham (2014) hat diese Struktur unter dem Begriff **Tidy Data** auf drei einfache Regeln gebracht:

- Eine **Variable** steht in genau einer Spalte.
- Eine **Beobachtung** steht in genau einer Zeile.
- Eine **Beobachtungseinheit** bildet genau eine Tabelle.

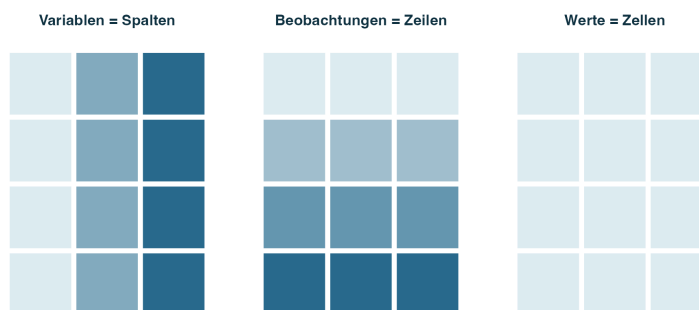


Abbildung 3.3: Tidy Data: In einem aufgeräumten Datensatz ist jede Variable eine Spalte, jede Beobachtung eine Zeile und jeder Wert eine Zelle (Wickham, 2014).

So selbstverständlich das klingt, so oft ist es in realen Datensätzen verletzt – etwa wenn mehrere Messzeitpunkte in nebeneinanderstehenden Spalten liegen oder eine Zelle gleich zwei Informationen enthält. Tidy Data ist die Struktur, in der (fast) alle Analysen und Grafiken in R erwartet werden; das **tidyverse** verdankt ihr sogar seinen Namen. Einen Datensatz „tidy“, zu machen, ist deshalb häufig der erste Schritt jeder Aufbereitung.

Eng damit verbunden ist die Unterscheidung zwischen **wide-** und **long-**Format. Dieselben Daten lassen sich einmal breit (jeder Messzeitpunkt eine eigene Spalte) und einmal lang (eine Zeile pro Messung) dar-

stellen. Der Wechsel zwischen beiden Formaten mit `pivot_longer()` und `pivot_wider()` gehört zu den häufigsten Aufgaben im Datenmanagement überhaupt – wir kommen im praktischen Teil ausführlich darauf zurück.

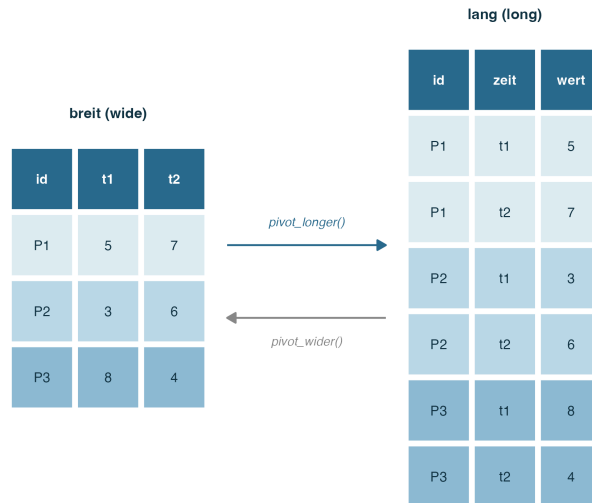


Abbildung 3.4: Dieselben Daten im breiten und im langen Format. `pivot_longer()` macht aus den Messzeitpunkt-Spalten Zeilen, `pivot_wider()` kehrt das um. Die Farbe markiert die Zugehörigkeit zur selben Person.

3.3 - Datenqualität und auffällige Daten

Ein großer Teil der praktischen Arbeit besteht darin, die Qualität der erhobenen Daten zu beurteilen und mit auffälligen Beobachtungen umzugehen. Dabei ist es hilfreich, sich klarzumachen, dass „auffällig“, nicht gleich „auffällig“ ist. Man kann sich ein **Spektrum** vorstellen, das von zufällig-harmlosen bis zu absichtlich-manipulierten Werten reicht: vom schlichten **Messfehler oder Ausreißer** über **unaufmerksames Antworten** bis hin zur bewussten **Fabrikation**. Das Tückische daran: Die *Werkzeuge*, mit denen man diese Fälle aufspürt, ähneln sich über das ganze Spektrum hinweg – die *Konsequenzen* für die Beteiligten könnten aber kaum unterschiedlicher sein.

3.3.1 - Unaufmerksames Antworten

Gerade in Online-Befragungen gibt es regelmäßig Teilnehmende, die den Fragebogen nicht ernsthaft bearbeiten, sondern nur durchklicken. Die Forschung zum **careless responding** (Curran, 2016; Meade & Craig, 2012) hat dafür einen ganzen Werkzeugkasten entwickelt, der weit über die naheliegenden Reaktionszeiten hinausgeht:

- **Attention Checks** (Instructed Manipulation Checks, IMC): Items, die eine bestimmte Antwort vorschreiben („Bitte kreuzen Sie hier *stimme voll zu an*“).
- **Longstring** / Straightlining: lange Serien identischer Antworten, wenn jemand stur dieselbe Kategorie ankreuzt.
- **Reaktionszeiten**: Antworten, die schlicht zu schnell kommen, um den Text überhaupt gelesen zu haben.

- **Mahalanobis-Distanz:** multivariate Ausreißer, deren Antwortmuster als Ganzes ungewöhnlich weit vom Rest entfernt liegt.
- **Konsistenzindizes:** etwa über psychometrische Antonyme, also inhaltlich gegenläufige Items, die konsistent beantwortet werden sollten.

Ein Blick auf die Verteilung der Reaktionszeiten (Abbildung 3.5) macht das Prinzip anschaulich. Der Hauptteil der Antworten bildet eine breite, rechtsschiefe Verteilung mit einem Gipfel bei realistischen Bearbeitungszeiten. Links davon, deutlich abgesetzt, sammelt sich eine kleine Gruppe extrem kurzer Zeiten – zu schnell, um die Frage gelesen zu haben.

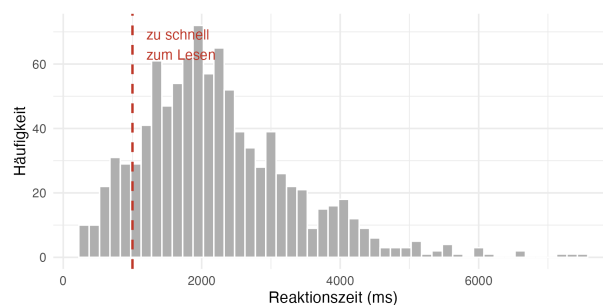


Abbildung 3.5: Verteilung der Reaktionszeiten mit einem plausiblen Ausschluss-Kriterium. Links der gestrichelten Linie liegen Antworten, die zu schnell für ein ernsthaftes Lesen sind.

3.3.2 - Fehlende Werte

Ein Sonderfall auffälliger Daten sind Werte, die gar nicht erst vorhanden sind. Für den Umgang mit fehlenden Werten ist entscheidend, *warum* sie fehlen. Man unterscheidet üblicherweise drei Mechanismen:

- **MCAR** (*missing completely at random*): rein zufällig fehlend, unabhängig von allem anderen.
- **MAR** (*missing at random*): fehlend in Abhängigkeit von *beobachteten* Variablen (z. B. antworten jüngere Personen seltener auf eine bestimmte Frage).
- **MNAR** (*missing not at random*): fehlend in Abhängigkeit vom fehlenden Wert *selbst* (z. B. verschweigen gerade Personen mit sehr hohem Einkommen ihr Einkommen).

Beim Umgang gilt: Das einfache Löschen betroffener Fälle (listwise oder pairwise deletion) ist nur unter MCAR unproblematisch; andernfalls verzerrt es die Ergebnisse. Anspruchsvollere Verfahren wie die **multiple Imputation** schätzen die fehlenden Werte aus den übrigen. In jedem Fall aber gilt eine Grundregel: Das **Muster der fehlenden Werte** gehört berichtet.

3.3.3 - Die Kehrseite der vielen Schritte

Bis hierhin klang das Aufspüren und Ausschließen auffälliger Daten wie reine Datenhygiene. Es hat jedoch eine gefährliche Kehrseite. Genau dieselben Werkzeuge, mit denen man Daten *sauber* macht, lassen sich nutzen, um Ergebnisse in die gewünschte Richtung zu verzerren. Ein Ausschlusskriterium, das man erst *nach* dem Blick auf die Ergebnisse wählt – „ohne diese drei Ausreißer wird der Effekt signifikant“, –,

ist ein sogenannter **researcher degree of freedom** (Simmons et al., 2011).

Das Problem verschärft sich, weil es selten nur *eine* solche Entscheidung gibt. Schließt man Ausreißer bei 2 oder bei 3 Standardabweichungen aus? Nimmt man eine Kovariate mit auf oder nicht? Jede dieser plausiblen Weggabelungen führt zu einer etwas anderen Analyse. Andrew Gelman und Eric Loken (2014) haben dafür das Bild vom „**garden of forking paths**“, geprägt (Abbildung 3.6): Wer aus vielen gleichermaßen vertretbaren Analysepfaden im Nachhinein den auswählt, der ein signifikantes Ergebnis liefert, findet fast garantiert *irgendeinen* – auch dann, wenn in der Grundgesamtheit gar kein Effekt existiert.

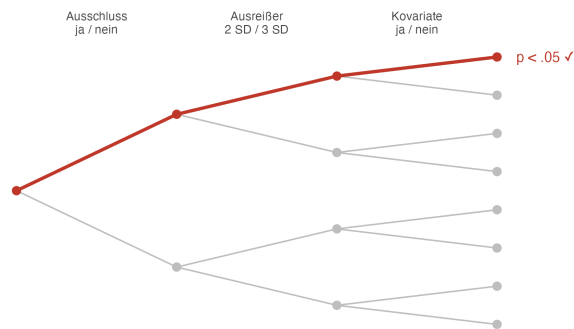


Abbildung 3.6: Der „garden of forking paths“: Aus vielen plausiblen Analyseentscheidungen wird im Nachhinein der eine Pfad ausgewählt, der zu $p < .05$ führt (rot).

⚠ Warnung

Der Ausweg besteht nicht darin, keine Ausreißer mehr auszuschließen, sondern darin, die Entscheidungen *vorab* festzulegen. Konkret heißt das: Ausschluss- und Aufbereitungsregeln **präregistrieren** (Nosek et al., 2018), bevor man die Ergebnisse kennt. Wo das nicht möglich ist, sollte man die **Robustheit** zeigen und das Ergebnis mit *und* ohne Ausschluss berichten (sensitivity- oder multiverse-Analyse). Kurz: **Transparenz schlägt Perfektion.**

3.3.4 - Extremfall: Datenforensik

Am äußersten Ende des Spektrums steht die absichtliche **Fabrikation** von Daten. Die beiden Fälle, mit denen wir dieses Kapitel gerahmt haben, markieren die Extreme: Reinhart-Rogoff war ein *ehrlicher* Fehler, sichtbar geworden durch geteilte Daten. Fabrikation ist eine *bewusste* Manipulation – die interessanterweise durch genau dieselbe Offenheit aufgedeckt wird.

Ein vieldiskutiertes Beispiel ist ironischerweise ein Fall um eine Studie zu Ehrlichkeit aus dem Jahr (2012), die auf Versicherungsdaten beruhte. Forscher, die das Blog *Data Colada* betreiben, [berichteten 2021](#), dass Sie in einem zugehörigen Datensatz sich ein verräterisches Muster zeigte: Ein Teil der Datensätze war offenbar **dupliziert** und die Kopien mit kleinen Zufallszahlen (1 bis 1000) minimal verändert worden. Der Originalteil war in der Schriftart **Calibri** gesetzt, die Kopie in **Cambria**. Und die in der Abbildung berichteten gefahrenen Meilen waren annä-

hernd **gleichverteilt** statt, wie man es erwarten würde, glockenförmig – als hätte ein Zufallsgenerator sie erzeugt (Abbildung 3.7).

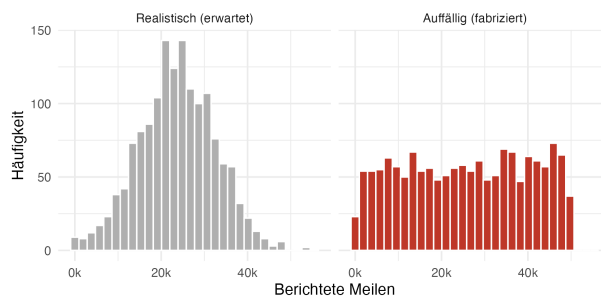


Abbildung 3.7: Die Verteilungsform als forensischer „Tell„: Echte Kilometerangaben sind glockenförmig verteilt (links), eine mit einem Zufallsgenerator erzeugte Variable dagegen annähernd gleichverteilt (rechts).

Aus solchen Fällen ist ein kleines Instrumentarium der **Datenforensik** entstanden. Der **GRIM-Test** (Brown & Heathers, 2016) etwa prüft, ob ein berichteter Mittelwert bei ganzzahligen Daten und gegebener Stichprobengröße überhaupt möglich ist. **Benford's Law** macht Aussagen über die zu erwartende Verteilung der führenden Ziffern in vielen natürlichen Datensätzen. Und oft sind es die einfachsten Auffälligkeiten – **Duplikate, Metadaten, die Form der Verteilung** –, die als Erstes stutzig machen. All diese Werkzeuge haben jedoch eine gemeinsame, unabdingbare Voraussetzung: den **Zugang zu den Rohdaten**. Damit schließt sich der Kreis zum Anfang des Kapitels.

3.4 - Datenmanagement mit R

Nachdem geklärt ist, *warum* jeder Aufbereitungsschritt dokumentiert und reproduzierbar sein sollte, geht es nun darum, *wie* das praktisch funktioniert. Wir arbeiten dazu durchgängig mit dem **tidyverse**, einer Sammlung von R-Paketen, die genau für diese Aufgaben gemacht ist. Bevor wir die einzelnen Schritte durchgehen, klären wir drei Dinge, die Sie in jedem Schritt brauchen: den Ablauf, das Grundmuster der Funktionen und die Pipe.

3.4.1 - Beispieldatensätze

Wir werden beispielhaft wieder mit dem Datensatz zu den Pinguinen der Palmer-Station in der Antarktis arbeiten, den Sie aus dem vorigen Kapitel kennen. Diesmal beginnen wir aber nicht mit der aufgeräumten Version **penguins**, sondern mit den **Rohdaten penguins_raw**, so wie sie im Feld erhoben wurden: mit unhandlichen Spaltennamen, zusätzlichen Variablen, Kommentaren und fehlenden Werten. Aus diesen Rohdaten bauen wir Schritt für Schritt einen analysefertigen Datensatz.

```
library(tidyverse)
library(palmerpenguins)
```

Die Rohdaten **penguins_raw** enthalten 344 Tiere und 17 Variablen, darunter Körpermaße, Isotopenwerte aus Blutproben, das Datum der Eiablage und Kommentare der Forschenden.

3.4.2 - Übersicht über den Ablauf

Rohdaten sind read-only.
Jede Veränderung entsteht als neues Objekt im Skript.

Der Weg von den Rohdaten zum analysefertigen Datensatz folgt fast immer demselben Muster. Ausgangspunkt sind die **Rohdaten**, die wir grundsätzlich als *read-only* behandeln. Von dort geht es über eine Kette klar benennbarer Schritte, die auch die Gliederung dieses Abschnitts bilden:

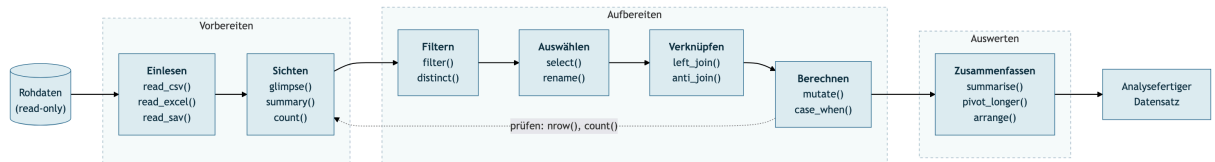


Abbildung 3.8: Die Schritte des Datenmanagements und die zugehörigen tidyverse-Verben. Nach jedem Schritt wird geprüft, ob das Ergebnis den Erwartungen entspricht.

Die Reihenfolge der Schritte wird bei den meisten Analysen recht viel Sinn ergeben. Wer zuerst filtert und auswählt, arbeitet in allen folgenden Schritten mit einem kleineren, übersichtlicheren Datensatz. Wer vor dem Berechnen verknüpft, hat alle Variablen beisammen, die er für neue Variablen braucht. Und das Zusammenfassen steht am Ende, weil dabei Information verloren geht: Aus vielen Zeilen wird eine Kennzahl. Es kann aber sein, dass man diese Schritte in einer anderen Reihenfolge leichter umsetzen kann. Entscheidend ist in jedem Fall, dass jeder dieser Schritte im Skript steht und nichts von Hand geändert wird.

3.4.3 - Pipes

`|>` liest man als „und dann„.
Tastenkürzel in RStudio: Strg + Shift + M (Mac: Cmd + Shift + M).

Das Bindeglied dieser Ketten ist die **Pipe** `|>`. Sie liest sich schlicht als „und dann„, und übergibt das Ergebnis der linken Seite als erstes Argument an die Funktion rechts. Statt Funktionsaufrufe ineinander zu verschachteln, was man von innen nach außen lesen muss, entsteht so eine Folge von Schritten, die man von oben nach unten liest. Dieselbe Auswertung einmal verschachtelt und einmal mit Pipe:

```
# verschachtelt: von innen nach außen zu lesen
summarise(
  group_by(
    filter(penguins, !is.na(body_mass_g)),
    species),
  m = mean(body_mass_g))

# mit Pipe: von oben nach unten zu lesen
penguins |>
  filter(!is.na(body_mass_g)) |>
  group_by(species) |>
  summarise(m = mean(body_mass_g))
```

Beide Varianten erzeugen exakt dasselbe Ergebnis, die zweite ist aber ungleich leichter zu lesen und zu erweitern. In RStudio fügt das Tastenkürzel **Strg + Shift + M** eine Pipe ein. In älteren Skripten und Foren begegnet Ihnen häufig noch die Pipe `%>%` aus dem Paket `magrittr`; für unsere Zwecke funktioniert sie genauso.

3.5 - Die einzelnen Schritte des Datenmanagements mit R

Der Kern der Aufbereitung im `tidyverse` sind einige wenige Funktionen, die man sich am besten als **Verben** merkt. Jedes tut genau eine Sache:

- `read_csv()` und Verwandte: Daten **einlesen**
- `glimpse()`, `summary()`, `count()`: Daten **sichten**
- `filter()`: **Zeilen** auswählen
- `select()`: **Spalten** auswählen
- `*_join()`: Tabellen **verknüpfen**
- `mutate()`: Variablen **berechnen**
- `summarise()` und `arrange()`: **zusammenfassen** und sortieren

Die Verben zur Bearbeitung folgen alle demselben Muster: `verb(daten, was_getan_wird)`. Das erste Argument ist immer der Datensatz, der Rückgabewert ist wieder ein Datensatz, und der Originaldatensatz bleibt dabei stets **unverändert**. Wer eine Änderung behalten möchte, muss das Ergebnis mit `<-` einem Objekt zuweisen, am besten einem neuen. Diese Einheitlichkeit ist das Konstruktionsprinzip des `tidyverse` und der Grund, warum sich die Verben so mühelos zu Ketten verbinden lassen.

Grundmuster aller Verben:
`verb(daten, was_getan_wird)`.
 Rein kommt ein Datensatz, raus kommt ein Datensatz.

3.5.1 - Einlesen

Am Anfang jeder Auswertung steht das Einlesen der Daten in R. Für die gängigen Dateiformate gibt es jeweils eine passende Funktion:

- `read_csv()` (Paket `readr`, Teil des `tidyverse`) für kommagetrennte Textdateien; für deutsche CSV-Dateien mit Semikolon als Trennzeichen und Komma als Dezimalzeichen gibt es `read_csv2()`.
- `read_excel()` (Paket `readxl`) für Excel-Dateien, mit dem Argument `sheet` für das gewünschte Tabellenblatt.
- `read_sav()` (Paket `haven`) für SPSS-Dateien, wie sie viele Befragungplattformen exportieren. Dabei bleiben die Wertelabels erhalten.

Das Paket `palmerpenguins` liefert die Rohdaten auch als CSV-Datei mit. Mit `path_to_file()` erhalten wir den Pfad zu dieser Datei und können sie so einlesen, wie Sie später Ihre eigenen Daten einlesen werden:

```
penguin_roh
<- read_csv(path_to_file("penguins_raw.csv"),
             show_col_types = FALSE)
```

Zwei Gewohnheiten ersparen Ihnen beim Einlesen viel Ärger. Erstens: Arbeiten Sie in einem **RStudio-Projekt** und geben Sie Pfade relativ zum Projektordner an, also `read_csv("daten/roh/befragung.csv")` statt `read_csv("C:/Users/.../befragung.csv")`. Dann läuft das Skript auch auf dem Rechner einer Kollegin. Zweitens: Legen Sie die Rohdaten in einen eigenen Ordner, den Sie nie beschreiben. Alles, was Ihr Skript erzeugt, speichern Sie woanders, etwa mit `write_csv(daten, "daten/aufbereitet/penguin.csv")`.

Relative Pfade beziehen sich auf den Projektordner und funktionieren deshalb auf jedem Rechner. **Absolute Pfade** wie `C:/Users/...` funktionieren nur auf Ihrem.

i Hinweis

Beim Einlesen rät `read_csv()` anhand der ersten Zeilen, welchen Datentyp jede Spalte hat. Das geht meistens gut, aber nicht immer: Steht in einer Zahlenspalte irgendwo ein Text wie "k.A." oder ".", wird die ganze Spalte als Text eingelesen. Deshalb folgt auf das Einlesen immer das Sichten.

3.5.2 - Sichten

Direkt nach dem Einlesen, sollte man sich einen Überblick verschaffen. Die wichtigste Funktion dafür ist `glimpse()`. Sie zeigt die Zahl der Zeilen und Spalten sowie für jede Spalte den Datentyp und die ersten Werte:

```
glimpse(pinguine_roh)
```

```
Rows: 344
Columns: 17
$ studyName      <chr> "PAL0708", "PAL0708",
"PAL0708", "PAL0708", "PAL...
$ `Sample Number` <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9,
10, 11, 12, 13, 14, 1...
$ Species        <chr> "Adelie Penguin (Pygoscelis
adeliae)", "Adelie P...
$ Region         <chr> "Anvers", "Anvers", "Anvers",
"Anvers", "Anvers"...
$ Island         <chr> "Torgersen", "Torgersen",
"Torgersen", "Torgerse...
$ Stage          <chr> "Adult, 1 Egg Stage", "Adult,
1 Egg Stage", "Adu...
$ `Individual ID` <chr> "N1A1", "N1A2", "N2A1",
"N2A2", "N3A1", "N3A2", ...
$ `Clutch Completion` <chr> "Yes", "Yes", "Yes", "Yes",
"Yes", "Yes", "No", ...
$ `Date Egg`     <date> 2007-11-11, 2007-11-11,
2007-11-16, 2007-11-16,...
$ `Culmen Length (mm)` <dbl> 39.1, 39.5, 40.3, NA, 36.7,
39.3, 38.9, 39.2, 34...
$ `Culmen Depth (mm)` <dbl> 18.7, 17.4, 18.0, NA, 19.3,
20.6, 17.8, 19.6, 18...
$ `Flipper Length (mm)` <dbl> 181, 186, 195, NA, 193, 190,
181, 195, 193, 190,...
$ `Body Mass (g)` <dbl> 3750, 3800, 3250, NA, 3450,
3650, 3625, 4675, 34...
$ Sex           <chr> "MALE", "FEMALE", "FEMALE",
NA, "FEMALE", "MALE"...
$ `Delta 15 N (o/oo)` <dbl> NA, 8.95, 8.37, NA, 8.77,
8.66, 9.19, 9.46, NA, ...
$ `Delta 13 C (o/oo)` <dbl> NA, -24.7, -25.3, NA, -25.3,
-25.3, -25.2, -24.9...
$ Comments      <chr> "Not enough blood for
isotopes.", NA, NA, "Adult..."
```

Schon dieser Blick verrät einiges. Der Datensatz hat mehr Spalten als die aufgeräumte Version, darunter Isotopenwerte aus Blutproben und eine Spalte mit Kommentaren. Die Spaltennamen enthalten Leer-

zeichen und Klammern, etwa `Body Mass (g)`. Solche Namen muss man in R in Backticks setzen (``Body Mass (g)``), was schnell mühsam wird; wir ändern das im Schritt Auswählen. Und statt „bill,“ für den Schnabel steht hier „Culmen“, der Fachbegriff für den Schnabelfirst.

Mit `summary()` erhält man für jede Spalte die wichtigsten Kennwerte. Nebenbei macht die Zeile `NA's` sichtbar, in welchen Variablen wie viele Werte fehlen:

```
pinguine_roh |>
  select(`Body Mass (g)`, `Flipper Length (mm)`, `Delta
15 N (o/oo)`) |>
  summary()
```

Body Mass (g)	Flipper Length (mm)	Delta 15 N (o/oo)
Min. :2700	Min. :172	Min. : 7.63
1st Qu.:3550	1st Qu.:190	1st Qu.: 8.30
Median :4050	Median :197	Median : 8.65
Mean :4202	Mean :201	Mean : 8.73
3rd Qu.:4750	3rd Qu.:213	3rd Qu.: 9.17
Max. :6300	Max. :231	Max. :10.03
NA's :2	NA's :2	NA's :14

Für kategoriale Variablen ist `count()` das Mittel der Wahl. Die Funktion zählt, wie oft jede Ausprägung vorkommt, und deckt dabei unerwartete Kodierungen, Tippfehler und fehlende Werte auf:

```
count(pinguine_roh, Species)
```

```
# A tibble: 3 × 2
  Species                                n
  <chr>                                <int>
1 Adelie Penguin (Pygoscelis adeliae) 152
2 Chinstrap penguin (Pygoscelis antarctica) 68
3 Gentoo penguin (Pygoscelis papua) 124
```

```
count(pinguine_roh, Sex)
```

```
# A tibble: 3 × 2
  Sex      n
  <chr> <int>
1 FEMALE 165
2 MALE   168
3 <NA>    11
```

Besonders aufschlussreich sind oft Freitextspalten wie die Kommentare. Hier haben die Forschenden notiert, wenn bei einem Tier etwas Besonderes vorfiel, zum Beispiel zu wenig Blut für die Isotopenanalyse. Solche Hinweise sind ein Stück Dokumentation, das bei der Entscheidung über Ausschlüsse hilft:

```
count(pinguine_roh, Comments, sort = TRUE)
```

```
# A tibble: 11 × 2
  Comments                                n
  <chr>                                <int>
1 <NA>                                290
2 Nest never observed with full clutch.
```

`dbl` steht für Kommazahlen, `chr` für Text, `date` für Datumsangaben. Steht bei einer Zahlenvariable `chr`, stimmt beim Einlesen etwas nicht.

```

34      3      Not      enough      blood      for      isotopes.
7      4      Sexing      primers      did      not      amplify.
4      5      No      blood      sample      obtained      for      sexing.
2      6      No      blood      sample      obtained.
2      7      Adult      not      sampled.
1      8 Adult not sampled. Nest never observed with full clutch.
1      9 Nest never observed with full clutch. Not enough blood
for isotopes.      1
10     No      delta15N      data      received      from      lab.
1      11 Sexing primers did not amplify. Not enough blood for
isotopes.      1

```

Zum freien Stöbern öffnet `View(pinguine_roh)` (großgeschrieben!) den Datensatz als sortier- und filterbare Tabelle in RStudio.

i Hinweis

`View()` ist ausschließlich zum *Anschauen* gedacht. Was Sie in dieser Tabellenansicht sortieren oder filtern, verändert die Daten **nicht** und wird auch nicht dokumentiert. Jede echte Änderung gehört ins Skript.

3.5.3 - Filtern

`filter()` behält genau die Zeilen, für die eine Bedingung `TRUE` ergibt. Als Bausteine dienen die Vergleichsoperatoren (`==`, `!=`, `<`, `>`, `<=`, `>=`) und die logischen Verknüpfungen `&` (und), `|` (oder) sowie `!` (nicht). Mehrere durch Komma getrennte Bedingungen werden automatisch mit *und* verknüpft. Mit `%in%` prüft man, ob ein Wert in einer Liste zulässiger Werte vorkommt:

```

pinguine_roh |>
  filter(Island %in% c("Biscoe", "Dream"), `Body Mass (g)`
> 4000) |>
  nrow()

```

```
[1] 161
```

Mit `|` lassen sich Bedingungen mit *oder* verknüpfen. Gesucht sind alle Tiere, die auf der Insel Torgersen leben *oder* eine Flosse von mehr als 220 mm haben:

```

pinguine_roh |>
  filter(Island == "Torgersen" | `Flipper Length (mm)` >
220) |>
  count(Island)

```

```

# A tibble: 2 × 2
  Island      n

```

	<chr>	<int>
1	Biscoe	35
2	Torgersen	52

Zwei Stolperfallen sind so häufig, dass man sie kennen sollte. Erstens heißt der Vergleich `==`, nicht `=`; das einfache Gleichheitszeichen führt zu einer Fehlermeldung. Zweitens, und tückischer: Ein fehlender Wert (`NA`) ist bei keinem Vergleich `TRUE`. Zeilen mit fehlenden Werten fallen bei einem Filter also **stillschweigend** heraus. Bei einigen Pinguinen konnte das Geschlecht nicht bestimmt werden. Filtert man nach weiblichen und nach männlichen Tieren, ergeben beide zusammen nicht die Gesamtzahl:

```
nrow(pinguine_roh)
```

```
[1] 344
```

```
pinguine_roh |> filter(Sex == "FEMALE") |> nrow()
```

```
[1] 165
```

```
pinguine_roh |> filter(Sex == "MALE") |> nrow()
```

```
[1] 168
```

```
pinguine_roh |> filter(is.na(Sex)) |> nrow()
```

```
[1] 11
```

Wer fehlende Werte gezielt ansprechen will, nutzt deshalb `is.na()` bzw. `!is.na()`.

Weil `filter()` das zentrale Werkzeug für den **Ausschluss von Fällen** ist, gelten hier die Prinzipien aus dem konzeptuellen Teil ganz unmittelbar: Das Kriterium sollte vorab feststehen, das Ergebnis kommt in ein *neues* Objekt, und die Zahl der ausgeschlossenen Fälle gehört berichtet. In unserem Beispiel schließen wir Tiere aus, bei denen keine Körpermaße vorliegen:

```
pinguine <- pinguine_roh |>
  filter(!is.na(`Body Mass (g)`))

nrow(pinguine_roh) - nrow(pinguine)      # Anzahl
ausgeschlossener Fälle
```

```
[1] 2
```

Ein Sonderfall des Filterns ist das Entfernen von **Duplikaten**. `distinct()` behält von mehreren identischen Zeilen nur eine. Gerade wenn Daten aus mehreren Exporten zusammengesetzt wurden, lohnt vorher ein Blick darauf, ob es Duplikate gibt, denn wie der Fall der Versicherungsdaten gezeigt hat, können sie auch ein Warnsignal sein.

3.5.4 - Auswählen

Was `filter()` für Zeilen ist, ist `select()` für Spalten. Man kann Spalten namentlich nennen, einzelne mit einem Minus ausschließen oder ganze

NA == irgendwas ist nie TRUE. Deshalb verschwinden Zeilen mit fehlenden Werten bei jedem Filter stillschweigend.

Ausgeschlossene Fälle gehören in den Methodenteil: Wie viele, und nach welchem vorab festgelegten Kriterium?

Bereiche mit einem Doppelpunkt angeben. Besonders praktisch ist, dass `select()` Spalten gleichzeitig **umbenennen** kann: Links vom `=` steht der neue, rechts der alte Name. So lösen wir im selben Schritt das Problem der unhandlichen Namen:

```
pinguine <- pinguine |>
  select(
    studie      = studyName,
    nr          = `Sample Number`,
    art_lang    = Species,
    insel       = Island,
    schnabel_mm = `Culmen Length (mm)`,
    flosse_mm   = `Flipper Length (mm)`,
    masse_g     = `Body Mass (g)`,
    geschlecht  = Sex,
    d15n        = `Delta 15 N (o/oo)`
  )

glimpse(pinguine)
```

```
Rows: 342
Columns: 9
$ studie      <chr> "PAL0708", "PAL0708", "PAL0708",
"PAL0708", "PAL0708", "PA...
$ nr          <dbl> 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12,
13, 14, 15, 16, 17, 18...
$ art_lang    <chr> "Adelie Penguin (Pygoscelis adeliae)",
"Adelie Penguin (Py...
$ insel       <chr> "Torgersen", "Torgersen", "Torgersen",
"Torgersen", "Torge...
$ schnabel_mm <dbl> 39.1, 39.5, 40.3, 36.7, 39.3, 38.9,
39.2, 34.1, 42.0, 37.8...
$ flosse_mm   <dbl> 181, 186, 195, 193, 190, 181, 195,
193, 190, 186, 180, 182...
$ masse_g     <dbl> 3750, 3800, 3250, 3450, 3650, 3625,
4675, 3475, 4250, 3300...
$ geschlecht  <chr> "MALE", "FEMALE", "FEMALE", "FEMALE",
"MALE", "FEMALE", "M...
$ d15n        <dbl> NA, 8.95, 8.37, 8.77, 8.66, 9.19,
9.46, NA, 9.13, 8.63, NA...
```

Links vom `=` steht immer der **neue** Name, rechts der alte. Das gilt für `select()`, `rename()` und `mutate()`.

Gute Variablennamen sind kurz, sprechend, klein geschrieben und enthalten keine Leerzeichen oder Sonderzeichen; die Einheit im Namen (`_mm`, `_g`) erspart spätere Rückfragen. Will man nur umbenennen und alle Spalten behalten, nimmt man `rename()` mit derselben Schreibweise.

Bei Datensätzen mit vielen ähnlich benannten Spalten, etwa Fragebögen mit durchnummerierten Items, sparen die **Hilfsfunktionen** viel Tipparbeit: `starts_with()`, `ends_with()`, `contains()` und `where()` wählen Spalten nach einem Muster oder einem Datentyp aus. In den Pinguin-Rohdaten beginnen etwa beide Isotopenwerte mit `Delta`, und alle Längenmaße enthalten die Einheit `(mm)`:

```
pinguine_roh |>
  select(starts_with("Delta")) |>
  names()
```

```
[1] "Delta 15 N (o/oo)" "Delta 13 C (o/oo)"
```

```
penguins_roh |>
  select(contains("(mm)")) |>
  names()
```

```
[1] "Culmen Length (mm)" "Culmen Depth (mm)" "Flipper
Length (mm)"
```

```
penguins_roh |>
  select(where(is.numeric)) |>
  names()
```

```
[1] "Sample Number"      "Culmen Length (mm)" "Culmen
Depth (mm)"
[4] "Flipper Length (mm)" "Body Mass (g)"      "Delta 15
N (o/oo)"
[7] "Delta 13 C (o/oo)"
```

Mit dem Minus lassen sich Spalten entfernen, die man nicht braucht. Die Spalten `Region` und `Stage` haben in den Rohdaten zum Beispiel für alle Tiere denselben Wert und tragen deshalb keine Information:

```
penguins_roh |>
  select(-c(Region, Stage)) |>
  ncol()
```

```
[1] 15
```

3.5.5 - Verknüpfen

In der Praxis liegen die Daten fast nie in einer einzigen Tabelle. Fragebogendaten, Reaktionszeiten, demografische Angaben und Laborwerte kommen aus verschiedenen Quellen und müssen zusammengeführt werden. Verknüpft wird über einen **Schlüssel** (key), also eine Spalte, die in beiden Tabellen vorkommt und dieselbe Beobachtungseinheit bezeichnet, typischerweise eine Personen-ID.

Als Beispiel nehmen wir an, eine Kollegin hat uns eine kleine Tabelle mit den deutschen Namen der drei Pinguinarten geschickt, die wir für Abbildungen verwenden wollen. In ihrer Tabelle heißen die Arten schlicht "Adelie", "Chinstrap" und "Gentoo":

```
arten <- tribble(
  ~art,      ~art_de,
  "Adelie",  "Adeliepinguin",
  "Chinstrap", "Zügelpinguin",
  "Gentoo",  "Eselpinguin"
)
```

Bevor man verknüpft, lohnt sich eine Probe mit `anti_join()`. Die Funktion gibt die Zeilen der linken Tabelle zurück, die rechts **keinen** Partner finden. Im Idealfall ist das Ergebnis leer:

```
penguins |>
  anti_join(arten, by = c("art_lang" = "art")) |>
  nrow()
```

```
[1] 342
```

Ein **Schlüssel** (key) identifiziert jede Beobachtungseinheit eindeutig, etwa eine Personen-ID oder ein Artnamen.

Kein einziges Tier findet einen Partner. Der Grund ist die häufigste Stolperfalle beim Verknüpfen: Die **Schlüssel passen nicht zusammen**. In den Rohdaten steht "Adelie Penguin (Pygoscelis adeliae)", in der Tabelle der Kollegin nur "Adelie". Wir bilden deshalb einen passenden Schlüssel, indem wir mit `word()` das erste Wort des langen Namens herausziehen. Dabei greifen wir auf `mutate()` vor, das wir im nächsten Abschnitt genauer besprechen:

```
penguin <- penguin |>
  mutate(art = word(art_lang, 1))

penguin |>
  anti_join(arten, by = "art") |>
  nrow()
```

```
[1] 0
```

Jetzt passen alle Schlüssel, und wir können verknüpfen. Der Standardfall ist der `left_join()`. Er behält **alle** Zeilen der linken Tabelle und ergänzt die passenden Spalten von rechts; fehlt ein Schlüssel rechts, entsteht an dieser Stelle ein `NA`. Der große Vorteil: Die Fallzahl der linken Tabelle bleibt erhalten, solange der Schlüssel in der rechten Tabelle eindeutig ist.

```
n_vorher <- nrow(penguin)

penguin <- penguin |>
  left_join(arten, by = "art", relationship = "many-to-one")

n_vorher; nrow(penguin)
```

```
[1] 342
```

```
[1] 342
```

```
count(penguin, art, art_de)
```

```
# A tibble: 3 × 3
  art      art_de      n
<chr>   <chr>    <int>
1 Adelie Adeliepinguin  151
2 Chinstrap Zügelpinguin   68
3 Gentoo  Eselspinguin  123
```

Das Argument `relationship = "many-to-one"` ist eine kleine Versicherung: Es teilt R mit, dass zu vielen Pinguinen jeweils genau eine Zeile in der Artentabelle gehören soll. Stünde eine Art dort versehentlich zweimal, würden sich die Zeilen beim Verknüpfen **unbemerkt vervielfachen**; mit dem Argument bricht R stattdessen mit einer Fehlermeldung ab.

Der `left_join()` ist nur das häufigste Mitglied einer ganzen **Join-Familie**, die sich darin unterscheidet, welche Zeilen sie behält:

- `left_join()`: alle Zeilen der linken Tabelle
- `inner_join()`: nur Zeilen mit Übereinstimmung in beiden Tabellen
- `full_join()`: alle Zeilen aus beiden Tabellen
- `anti_join()`: nur die Zeilen der linken Tabelle **ohne** Partner rechts

Zusammengefasst lauern beim Verknüpfen drei typische Probleme: Schlüssel, die sich in Schreibweise oder Format unterscheiden ("P07" vs. "P007"), Schlüssel, die rechts nicht eindeutig sind, und Schlüsselspalten, die in beiden Tabellen unterschiedlich heißen, was man wie oben mit `by = c("links" = "rechts")` auflöst. Die Faustregel lautet deshalb: vor jedem Join mit `anti_join()` prüfen und nach jedem Join `nrow()` vergleichen.

Faustregel: vor jedem Join `anti_join()`, nach jedem Join `nrow()`.

3.5.6 - Berechnen

`mutate()` legt neue Variablen an oder überschreibt bestehende. Der Name der neuen Variable steht **links** vom `=`, rechts steht eine Berechnung, die sich auf beliebige andere Spalten beziehen kann. Mehrere Berechnungen lassen sich in einem Aufruf erledigen, und spätere Berechnungen können auf frühere zugreifen:

```
pinguine |>
  mutate(
    masse_kg = masse_g / 1000,
    flosse_pro_kg = flosse_mm / masse_kg
  ) |>
  select(art, masse_g, masse_kg, flosse_pro_kg) |>
  head(3)
```

```
# A tibble: 3 × 4
  art    masse_g masse_kg flosse_pro_kg
<chr>   <dbl>   <dbl>       <dbl>
1 Adelie  3750     3.75         48.3
2 Adelie  3800     3.8          48.9
3 Adelie  3250     3.25          60
```

Eine besonders häufige Aufgabe ist das **Umkodieren**. Für Variablen mit zwei Ausprägungen genügt `if_else()`, für mehrere Ausprägungen ist `case_when()` das Mittel der Wahl. Die Bedingungen werden von oben nach unten geprüft, die erste zutreffende gewinnt, und `.default` fängt alles Übrige ab. So übersetzen wir die Geschlechtsangabe; alle anderen Einträge, auch fehlende oder unerwartete, werden zu `NA`:

`case_when()` prüft die Bedingungen **von oben nach unten**. Die erste zutreffende Bedingung gewinnt.

```
pinguine <- pinguine |>
  mutate(
    geschlecht = case_when(
      geschlecht == "FEMALE" ~ "weiblich",
      geschlecht == "MALE"   ~ "männlich",
      .default = NA
    )
  )

count(pinguine, geschlecht)
```

```
# A tibble: 3 × 2
  geschlecht    n
<chr>       <int>
1 männlich    168
2 weiblich    165
3 <NA>         9
```

Mit `case_when()` lassen sich auch **kontinuierliche Variablen in Kategorien** einteilen. Wir teilen die Pinguine nach ihrem Körpergewicht in drei Gewichtsklassen ein. Weil die Bedingungen von oben nach

unten geprüft werden, genügt für jede Klasse die obere Grenze. Eine Zeile, auf die keine Bedingung zutrifft, etwa weil der Wert fehlt, erhält ohne `.default` automatisch NA:

```
pinguine <- pinguine |>
  mutate(
    gewichtsklasse = case_when(
      masse_g < 3500 ~ "leicht",
      masse_g < 4500 ~ "mittel",
      masse_g >= 4500 ~ "schwer"
    )
  )

count(pinguine, gewichtsklasse)
```

```
# A tibble: 3 × 2
  gewichtsklasse     n
  <chr>          <int>
1 leicht         71
2 mittel        153
3 schwer        118
```

Weil Texte in Tabellen und Grafiken alphabetisch sortiert werden, sollte man bei kategorialen Variablen die inhaltlich sinnvolle Reihenfolge festlegen, indem man sie mit `factor()` und definierten `levels` in einen **Faktor** umwandelt:

```
pinguine <- pinguine |>
  mutate(gewichtsklasse = factor(gewichtsklasse,
    levels = c("leicht", "mittel", "schwer")))

levels(pinguine$gewichtsklasse)
```

```
[1] "leicht" "mittel" "schwer"
```

Ein z-Wert von 1 bedeutet: eine Standardabweichung über dem Mittelwert der eigenen Gruppe.

Manchmal soll eine Berechnung **innerhalb von Gruppen** erfolgen. Ein typisches Beispiel ist die z-Standardisierung, bei der man jeden Wert am Mittelwert und an der Standardabweichung seiner eigenen Gruppe relativiert. Mit dem Argument `.by` berechnet `mutate()` Mittelwert und Standardabweichung getrennt für jede Art, und das Ergebnis sagt dann, ob ein Pinguin *für seine Art* schwer oder leicht ist:

```
pinguine <- pinguine |>
  mutate(z_masse = (masse_g - mean(masse_g)) /
    sd(masse_g), .by = art)

pinguine |>
  select(art, masse_g, z_masse) |>
  head(3)
```

```
# A tibble: 3 × 3
  art    masse_g z_masse
  <chr>    <dbl>   <dbl>
1 Adelie  3750    0.108
2 Adelie  3800    0.217
3 Adelie  3250   -0.983
```

💡 Skalenwerte bei Fragebögen

In der Psychologie besteht das Berechnen häufig darin, aus mehreren Items einen Skalenwert zu bilden. Dafür kombiniert man `rowMeans()` mit `across()`, das dieselben Auswahlhilfen versteht wie `select()`. Negativ gepolte Items dreht man vorher um, bei einer Skala von 1 bis 5 etwa mit `6 - item`:

```
befragung |>
  mutate(
    item_03 = 6 - item_03,      # negativ gepoltes
    Item umpolen
    skala = rowMeans(across(starts_with("item")),
    na.rm = TRUE)
  )
```

3.5.7 - Zusammenfassen

Um von Einzelbeobachtungen zu Kennwerten zu gelangen, fasst man die Daten zusammen. `summarise()` berechnet aus vielen Zeilen eine Ergebniszeile; mit dem Argument `.by` geschieht das **pro Gruppe**. Die Funktion `n()` zählt dabei die Fälle je Gruppe:

```
penguins |>
  summarise(
    n = n(),
    m_masse = mean(masse_g),
    sd_masse = sd(masse_g),
    .by = art_de
  )
```

```
# A tibble: 3 × 4
  art_de      n m_masse sd_masse
<chr>    <int> <dbl>   <dbl>
1 Adeliepinguin  151  3701.   459.
2 Eselspinguin  123  5076.   504.
3 Zügelpinguin   68  3733.   384.
```

Dieselbe Auswertung finden Sie in vielen Skripten in der älteren Schreibweise mit einem vorangestellten `group_by(art_de)`; das Ergebnis ist identisch. Zwei Details lohnen sich zu merken. Erstens liefert `mean()` als Ergebnis `NA`, sobald auch nur ein Wert fehlt; mit `na.rm = TRUE` werden fehlende Werte ignoriert. Bei den Pinguinen haben wir diese Fälle bereits ausgeschlossen, bei den Isotopenwerten fehlen aber noch einige. Zweitens lohnt es sich, bei jeder Zusammenfassung die Fallzahl mitzuberichten, denn Mittelwerte aus drei Beobachtungen sind wenig belastbar. Beides lässt sich an den Isotopenwerten zeigen. Wir fassen nach Art *und* Insel zusammen und sortieren mit `arrange()` aufsteigend nach der Fallzahl. Ohne `na.rm = TRUE` ist der Mittelwert in allen Gruppen mit fehlenden Werten `NA`, und die Gruppen sind sehr unterschiedlich groß:

```
penguins |>
  summarise(
    n = n(),
    m_d15n = mean(d15n),
```

Beim Zusammenfassen geht Information verloren: Aus vielen Zeilen wird eine. Deshalb steht dieser Schritt am Ende.

```
m_d15n_rm = mean(d15n, na.rm = TRUE),
  .by = c(art_de, insel)
) |>
  arrange(n)
```

```
# A tibble: 5 × 5
  art_de      insel      n m_d15n m_d15n_rm
<chr>      <chr> <int> <dbl> <dbl>
1 Adeliepinguin Biscoe    44  8.82  8.82
2 Adeliepinguin Torgersen  51  NA    8.79
3 Adeliepinguin Dream    56  NA    8.95
4 Zügelpinguin Dream    68  NA    9.36
5 Eselspinguin Biscoe   123  NA    8.25
```

Mit `desc()` sortiert `arrange()` absteigend, etwa `arrange(desc(n))`.

Wer nur zählen möchte, kommt mit `count()` aus, das wir schon beim Sichten verwendet haben; es lässt sich auch nach mehreren Variablen gleichzeitig aufschlüsseln:

```
count(pinguine, art_de, geschlecht)
```

```
# A tibble: 8 × 3
  art_de      geschlecht      n
<chr>      <chr> <int>
1 Adeliepinguin männlich    73
2 Adeliepinguin weiblich    73
3 Adeliepinguin <NA>         5
4 Eselspinguin männlich    61
5 Eselspinguin weiblich    58
6 Eselspinguin <NA>         4
7 Zügelpinguin männlich    34
8 Zügelpinguin weiblich    34
```

Damit kommen wir zu der im konzeptuellen Teil angekündigten Umformung zwischen **wide** und **long**. Möchte man mehrere Variablen auf einmal zusammenfassen, bringt man sie zuerst mit `pivot_longer()` in die lange Form: Aus den drei Maßspalten wird eine Spalte `mass` mit dem Namen des Maßes und eine Spalte `wert` mit dem Messwert. Danach genügt ein einziges `summarise()` für alle Maße:

```
pinguine_lang <- pinguine |>
  pivot_longer(
    cols = c(schnabel_mm, flosse_mm, masse_g),
    names_to = "mass",
    values_to = "wert"
  )

pinguine_lang |>
  summarise(m = mean(wert), .by = c(art_de, mass))
```

```
# A tibble: 9 × 3
  art_de      mass      m
<chr>      <chr> <dbl>
1 Adeliepinguin schnabel_mm  38.8
2 Adeliepinguin flosse_mm   190.
3 Adeliepinguin masse_g    3701.
4 Eselspinguin schnabel_mm  47.5
5 Eselspinguin flosse_mm   217.
```

Immer die Fallzahl mitberichten. Ein Mittelwert aus drei Beobachtungen ist wenig belastbar.

`pivot_longer()`: aus Spalten werden Zeilen.
`pivot_wider()`: aus Zeilen werden Spalten.

```
6 Eselspinguin masse_g      5076.
7 Zügelpinguin schnabel_mm  48.8
8 Zügelpinguin flosse_mm    196.
9 Zügelpinguin masse_g      3733.
```

Das Ergebnis ist tidy, aber als Tabelle für einen Bericht unpraktisch. `pivot_wider()` macht aus den Werten einer Spalte wieder eigene Spalten und ist damit die Umkehrung von `pivot_longer()`. So entsteht eine kompakte Tabelle mit einer Zeile pro Art und einer Spalte pro Maß:

```
pinguine_lang |>
  summarise(m = round(mean(wert), 1), .by = c(art_de,
mass)) |>
  pivot_wider(names_from = mass, values_from = m)
```

```
# A tibble: 3 × 4
  art_de      schnabel_mm flosse_mm masse_g
<chr>      <dbl>      <dbl>   <dbl>
1 Adeliepinguin      38.8        190    3701.
2 Eselspinguin       47.5        217    5076
3 Zügelpinguin       48.8        196    3733.
```

3.5.8 - Übersichtstabellen mit `compareGroups`

Für die in fast jeder Publikation übliche „Tabelle 1“, also alle wichtigen Variablen nach Gruppen aufgeschlüsselt mit passenden Kennwerten und Signifikanztests, muss man die Schritte aus dem vorigen Abschnitt nicht von Hand zusammenbauen. Das Paket `compareGroups` erledigt das in zwei Schritten: `compareGroups()` berechnet, `createTable()` formatiert. Die Auswahl der Variablen erfolgt über die Formel-Schreibweise **Gruppe ~ Variablen**:

Die „Tabelle 1“ beschreibt in fast jeder empirischen Arbeit die Stichprobe, meist getrennt nach Gruppen.

```
library(compareGroups)

compareGroups(art_de ~ masse_g + flosse_mm + geschlecht
+ d15n,
              data = pinguine) |>
  createTable(show.n = TRUE)
```

```
-----Summary descriptives table by 'art_de'-----
```

		Adeliepinguin	Eselspinguin	Zügelpinguin
p.overall	N	N=151	N=123	N=68
masse_g		3701 (459)	5076 (504)	3733 (384)
<0.001	342			
flosse_mm		190 (6.54)	217 (6.48)	196 (7.13)
<0.001	342			
geschlecht:				0.976 333
männlich	73 (50.0%)	61 (51.3%)	34 (50.0%)	
weiblich	73 (50.0%)	58 (48.7%)	34 (50.0%)	
d15n		8.86 (0.43)	8.25 (0.26)	9.36 (0.37)
<0.001	330			

Über weitere Optionen lässt sich die Tabelle anpassen, etwa `method = c(d15n = 2)` für Median und Interquartilsabstand statt Mittelwert und Standardabweichung. Für den Export gibt es `export2word()`, `export2latex()` und `export2md()`.

i Hinweis

`compareGroups` ist ein Zusatzpaket, das Sie ggf. einmalig mit `install.packages("compareGroups")` installieren müssen.

3.5.9 - Der ganze Workflow in einer Kette

Ein Skript, das von der Rohdatei bis zum Ergebnis durchläuft, ist die beste Dokumentation Ihrer Datenaufbereitung.

Zum Abschluss lassen sich alle Schritte zu einer einzigen, von oben nach unten lesbaren Kette verbinden. Genau darin liegt die Stärke des Ansatzes: ein Skript, in dem jeder Schritt sichtbar und reproduzierbar ist, von der Rohdatei bis zum analyserefertigen Datensatz.

```
penguin_analyse
<- read_csv(path_to_file("penguins_raw.csv"),
              show_col_types = FALSE) |>
# einlesen
filter(!is.na(`Body Mass (g)`) |>                                #
filtern
  select(art_lang = Species,                                     #
auswählen
        insel      = Island,
        flosse_mm  = `Flipper Length (mm)`,
        masse_g    = `Body Mass (g)`,
        geschlecht = Sex) |>
mutate(art = word(art_lang, 1)) |>
left_join(arten, by = "art",                                     #
verknüpfen
          relationship = "many-to-one") |>
mutate(                                                         # berechnen
  geschlecht = case_when(geschlecht == "FEMALE" ~
"weiblich",
                        geschlecht == "MALE"   ~ "männlich",
                        .default = NA),
  z_masse = (masse_g - mean(masse_g)) / sd(masse_g),
  .by = art
)

penguin_analyse |>                                             #
zusammenfassen
  summarise(n = n(), m_masse = mean(masse_g), .by =
c(art_de, geschlecht)) |>
  arrange(art_de, geschlecht)
```

```
# A tibble: 8 × 4
  art_de      geschlecht    n m_masse
  <chr>      <chr>      <int> <dbl>
1 Adeliepinguin männlich    73  4043.
2 Adeliepinguin weiblich    73  3369.
3 Adeliepinguin <NA>         5   3540
4 Eselspinguin  männlich    61  5485.
5 Eselspinguin  weiblich    58  4680.
6 Eselspinguin  <NA>         4  4588.
```

```
7 Zügelpinguin männlich 34 3939.
8 Zügelpinguin weiblich 34 3527.
```

Diese Kette ist die praktische Umsetzung von allem, was der konzeptuelle Teil gefordert hat: Die Rohdatei bleibt unberührt, das Ergebnis entsteht als neues Objekt, und jeder einzelne Schritt, vom Ausschluss über die Verknüpfung bis zur Umkodierung, steht dokumentiert im Skript und kann im Bericht nachvollziehbar beschrieben werden.

3.6 - Fazit

- Datenmanagement ist keine lästige Fleißaufgabe am Rande, sondern entscheidet über die **Reproduzierbarkeit** und damit über die Glaubwürdigkeit einer Studie. Der Fall Reinhart-Rogoff zeigt, wie ein einziger undokumentierter Handgriff eine ganze Debatte verzerren kann.
- Die tragenden Prinzipien sind schnell benannt: Rohdaten bleiben **unantastbar**, jede Transformation passiert im **Skript**, und die Zielstruktur ist **tidy** – eine Variable pro Spalte, eine Beobachtung pro Zeile.
- Auffällige Daten sollte man identifizieren und ggf. ausschließen – aber die Kriterien gehören **vorab** festgelegt, sonst wird aus Datenhygiene unversehens p-hacking.
- Die Aufbereitung folgt einer festen Abfolge: **einlesen** → **sichten** → **filtern** → **auswählen** → **verknüpfen** → **berechnen** → **zusammenfassen**. Im **tidyverse** gibt es für jeden Schritt ein eigenes Verb (`read_csv`, `glimpse`, `filter`, `select`, `*_join`, `mutate`, `summarise`), und über die **Pipe** `|>` verketteten sie sich zu einem lesbaren, von oben nach unten nachvollziehbaren Workflow.
- Prüfen Sie nach jedem Schritt, was Sie erwarten: `nrow()`, `count()` und `summary()` decken die meisten stillen Fehler sofort auf.

3.6.1 - Vertiefung

Wenn Sie tiefer in die praktische Datenaufbereitung einsteigen möchten, führt kein Weg an *R for Data Science* von Wickham, Çetinkaya-Rundel und Grolemund vorbei – insbesondere die frei verfügbaren Kapitel zu *Data transformation* und *Data tidying* decken genau die hier vorgestellten Verben vertiefend ab. Die konzeptuelle Grundlage des tidy-Ansatzes legt Wickhams ursprüngliche Arbeit zu *Tidy Data* dar (Wickham, 2014). Wer sich für die forensische Seite begeistert, findet auf dem Blog *Data Colada* (datacolada.org) die Originalanalysen zu den oben erwähnten Fällen; eine gut lesbare Einführung in die Fallstricke flexibler Datenauswertung bietet der Klassiker von Simmons, Nelson und Simonsohn (2011). Und wer seine Daten am Ende teilen möchte, orientiert sich an den FAIR-Prinzipien (M. D. Wilkinson et al., 2016) und an den Datenmanagement-Vorgaben der eigenen Förderorganisation.

Es gibt auch einen sehr interessante Podcast-Folge [Freakonomics No.163: The Data Sleuth Taking on Shoddy Science](#), in dem der Fall aufgearbeitet wird.

II ALM und Erweiterungen

4	ALM und Multiple lineare Regression	67
4.1	Das Allgemeine Lineare Modell	67
4.2	Wie die Parameter geschätzt werden	73
4.3	Varianzaufklärung	76
4.4	Annahmen des linearen Modells	79
4.5	Regression mit R	81
4.6	Schicke Tabellen mit modelsummary	93
4.7	Fazit	99
5	Varianzanalysen	101
5.1	Die Idee der Varianzzerlegung	101
5.2	Mehrfaktorielle Varianzanalyse und Interaktionen	103
5.3	Kovarianzanalyse (ANCOVA)	105
5.4	Varianzanalyse mit Messwiederholung	106
5.5	Prä-post-Designs mit zwei Gruppen	107
5.6	Fallzahlplanung	109
5.7	Varianzanalysen mit R	110
5.8	Fazit	124
6	Logistische Regression	125
7	Das LASSO und andere Shrinkage-Verfahren	127
8	Mehrebenenmodelle	129

4. ALM und Multiple lineare Regression

„...less is more, except of course for sample size.“ – Jacob Cohen (1990)

In diesem Kapitel lernen Sie das **Allgemeinen Linearen Modell** kennen, dass viele Verfahren, die Sie im Bachelor vielleicht als getrennte Werkzeuge kennengelernt haben – t-Test, Korrelation, Varianzanalyse –, eigentlich Spezialfälle ein und desselben Modells sind. Danach klären wir, was es bedeutet, dass die Parameter dieses Modells nach dem Prinzip der **Maximum-Likelihood-Schätzung** geschätzt werden. Anschließend geht es um die Frage, wie das Gesamtmodell und die einzelnen Parameter sinnvoll interpretiert werden können. Zum Abschluss des konzeptuellen Teils besprechen wir die **Annahmen** des Modells³. Wie in den vorigen Kapiteln ist die zweite Hälfte praktisch: Sie zeigt, wie Sie Regressionsmodelle in R schätzen, prüfen, visualisieren und mit dem Paket `modelsummary` in publikationsreife Tabellen überführen.

Der Name des Verfahrens ist eigentlich ein historischer Zufall. Francis Galton (1886) untersuchte gegen Ende des 19. Jahrhunderts, wie die Körpergröße von Eltern mit der ihrer erwachsenen Kinder zusammenhängt. Dabei fiel ihm auf, dass die Kinder sehr großer Eltern zwar überdurchschnittlich groß waren, aber im Mittel *weniger* extrem als ihre Eltern – und umgekehrt für sehr kleine Eltern. Galton nannte das Phänomen „regression towards mediocrity“, die Rückkehr zum Mittelmaß. Heute sprechen wir von der **Regression zur Mitte**, und sie ist ein Artefakt jeder nicht perfekten Korrelation. Die Methode, mit der Galton den Zusammenhang beschrieb, behielt den Namen, obwohl sie mit einem „Rückschritt“ nichts zu tun hat.

4.1 - Das Allgemeine Lineare Modell

4.1.1 - Daten = Modell + Fehler

Die Grundidee aller Verfahren in diesem und den folgenden Kapiteln lässt sich in einer einzigen Zeile zusammenfassen:

$$\text{Daten} = \text{Modell} + \text{Fehler} \quad (4.1)$$

Wir versuchen, die beobachteten Werte einer **abhängigen Variable** (auch Kriterium oder Outcome) y durch eine einfache, formale Regel aus einer oder mehreren **unabhängigen Variablen** (Prädiktoren) $x_1, x_2, \dots, x_k$ vorherzusagen. Was die Regel nicht erklären kann, landet im Fehler. Im **Allgemeinen Linearen Modell** (ALM) ist diese Regel eine gewichtete Summe der Prädiktoren:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \dots + \beta_k x_{ki} + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2) \quad (4.2)$$

Die **Regression zur Mitte** ist bis heute eine der häufigsten Fehlerquellen bei der Bewertung von Interventionen: Wer gezielt die schlechtesten Abteilungen eines Unternehmens coacht, wird beim nächsten Mal fast immer eine Verbesserung sehen – auch ganz ohne Coaching.

³There ain't no such thing as a free lunch!

Linear heißt das Modell, weil es *linear in den Parametern* ist: Die β s werden nur mit etwas multipliziert und aufaddiert. Die Prädiktoren selbst dürfen durchaus transformiert sein – auch $y = \beta_0 + \beta_1 x + \beta_2 x^2$ ist ein lineares Modell, obwohl es eine Kurve beschreibt.

Dabei ist y_i der Wert von Person (oder bei uns Pinguin) i , β_0 der **Achsenabschnitt** (Intercept), also der vorhergesagte Wert, wenn alle Prädiktoren null sind, und β_1 bis β_k sind die **Regressionsgewichte** (Steigungen). Der **Fehler** oder das **Residuum** ε_i ist die Abweichung des beobachteten vom vorhergesagten Wert $\hat{y}_i$. Von den Fehlern nehmen wir an, dass sie im Mittel null sind, unabhängig voneinander und normalverteilt mit einer für alle Beobachtungen gleichen Varianz σ^2 . Auf diese Annahmen kommen wir am Ende des konzeptuellen Teils ausführlich zurück.

Kompakter lässt sich das Modell in Matrixschreibweise notieren. Alle y -Werte stehen in einem Vektor $\mathbf{y}$, alle Prädiktorwerte – ergänzt um eine Spalte aus Einsen für den Achsenabschnitt – in der sogenannten **Designmatrix** $\mathbf{X}$:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \quad (4.3)$$

Diese Schreibweise sieht zunächst abstrakt aus, sie macht aber einen zentralen Punkt sichtbar: Das Modell interessiert sich nicht dafür, *was* in den Spalten von $\mathbf{X}$ steht. Das können kontinuierliche Messwerte sein, aber genauso gut Nullen und Einsen, die eine Gruppenzugehörigkeit kodieren. Genau deshalb ist das Modell *allgemein*.

i Obacht: Allgemein oder generalisiert?

Im Englischen werden leider zwei verschiedene Modellfamilien als *GLM* abgekürzt. Das **Allgemeine** Lineare Modell (*general linear model*) aus diesem Kapitel setzt normalverteilte Fehler voraus. Das **Generalisierte** Lineare Modell (*generalized linear model*) erweitert es auf andere Verteilungen, etwa für binäre Outcomes (logistische Regression) oder Häufigkeiten (Poisson-Regression). Wir behandeln es im Kapitel zur logistischen Regression. In R entspricht das dem Unterschied zwischen `lm()` und `glm()`.

4.1.2 - Regressionsgewichte in der multiplen Regression

In der einfachen Regression mit nur einem Prädiktor ist β_1 schlicht die Steigung der Geraden: Sie beschreibt um wie viele Einheiten sich der vorhergesagte Wert von y ändert, wenn x um eine Einheit steigt. In der **multiplen** Regression mit mehreren Prädiktoren ändert sich die Bedeutung auf subtile, aber entscheidende Weise. Das Gewicht β_j beschreibt nun die vorhergesagte Veränderung in y , wenn x_j um eine Einheit steigt **und alle anderen Prädiktoren konstant bleiben**. Man spricht von einem **partiellen** Regressionsgewicht.

Wie groß der Unterschied zwischen einfachem und partiellem Gewicht sein kann, zeigt ein Beispiel aus dem Pinguin-Datensatz. Betrachtet man alle Tiere gemeinsam, hängen Schnabellänge und Schnabeltiefe *negativ* zusammen: Längere Schnäbel sind flacher. Nimmt man die Art als zweiten Prädiktor in das Modell auf, kehrt sich das Vorzeichen dieses Gewichtes. Innerhalb jeder Art gilt: Je länger der Schnabel, desto tiefer ist er auch (Abbildung 4.1). Der negative Gesamtzusammenhang entsteht nur, weil die Gentoo-Pinguine lange, aber flache Schnäbel haben. Dieses Muster ist als **Simpson-Paradox** bekannt.

Die Regressionsgewichte einzelner Prädiktoren in einer multiplen Regression können sich stark ändern in Abhängigkeit davon welche anderen Prädiktoren Teil des Modells sind.

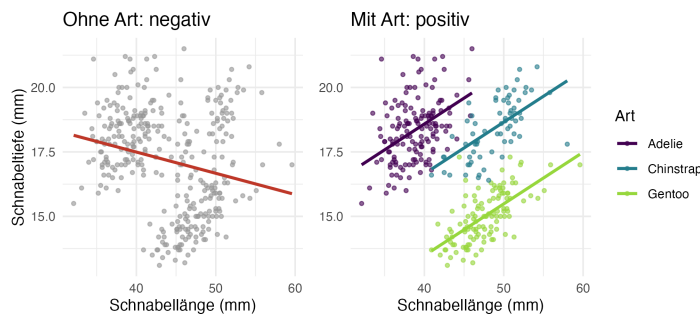


Abbildung 4.1: Simpson-Paradox bei den Palmer-Pinguinen. Links: Über alle Tiere hinweg hängen Schnabellänge und -tiefe negativ zusammen (einfache Regression). Rechts: Kontrolliert man die Art, zeigt sich innerhalb jeder Art ein positiver Zusammenhang (multiple Regression mit gleicher Steigung für alle Arten).

⚠ Statistische Kontrolle ist keine experimentelle Kontrolle

Die Formulierung „unter Konstanthaltung der anderen Prädiktoren,“ klingt nach Experiment, ist aber rein rechnerisch gemeint. In Beobachtungsdaten hält niemand etwas konstant; das Modell vergleicht lediglich Fälle, die sich in den übrigen Prädiktoren ähneln. Ob ein partielles Gewicht kausal interpretiert werden darf, hängt davon ab, ob die *richtigen* Variablen im Modell sind. Wer eine wichtige Störvariable vergisst, erhält verzerrte Gewichte; wer eine Variable aufnimmt, die selbst eine Folge von x und y ist (ein sogenannter *Collider*), erzeugt sogar Zusammenhänge, die es gar nicht gibt. Welche Variablen in ein Modell gehören, ist deshalb eine theoretische und keine statistische Frage.

Weil die Gewichte in den Einheiten der jeweiligen Variablen angegeben werden (Gramm pro Millimeter, Punkte pro Lebensjahr ...), lassen sie sich nicht direkt miteinander vergleichen. Standardisiert man vor der Schätzung alle Variablen (z-Werte), erhält man **standardisierte Regressionsgewichte** β^* : Sie geben an, um wie viele Standardabweichungen sich y verändert, wenn x_j um eine Standardabweichung steigt. Das erleichtert den Vergleich, hat aber einen Preis: Standardisierte Gewichte hängen von der Streuung der Variablen *in der jeweiligen Stichprobe* ab und lassen sich deshalb schlecht zwischen Studien vergleichen. Für kategoriale Prädiktoren sind sie ohnehin kaum sinnvoll interpretierbar.

In der Praxis ist deshalb ein Vorschlag von Gelman (2007) verbreitet: Man zentriert die kontinuierlichen Prädiktoren und teilt sie durch **zwei** Standardabweichungen, während binäre Prädiktoren (0/1) und das Kriterium in ihren ursprünglichen Einheiten bleiben. Das Gewicht gibt dann an, um wie viel sich y verändert, wenn man eine Person eine Standardabweichung unter dem Mittelwert mit einer Person eine Standardabweichung darüber vergleicht. Weil ein binärer-Prädiktor mit etwa gleich großen Gruppen eine Standardabweichung von 0,5 hat, entspricht ihr Sprung von 0 auf 1 ebenfalls zwei Standardabweichungen, sodass die Gewichte kontinuierlicher und binärer Prädiktoren recht gut vergleichbar werden. Die Tabelle unten stellt die drei Varianten gegenüber. Wichtig die unterschiedlichen Transformationen haben keinen

Einfluss auf die Gesamtmodellgüte, sondern stellen nur die Prädiktoren anders dar.

Tabelle 4.1: Regression des Körpergewichts auf Flossenlänge, Schnabellänge und Geschlecht mit drei Skalierungen. Unstandardisiert: Gramm pro Millimeter. Beta: Standardabweichungen des Körpergewichts pro Standardabweichung des Prädiktors. 2 SD: Gramm pro zwei Standardabweichungen des Prädiktors.

	Unstandardisiert	Beta (z-Werte)	2 SD
Flossenlänge	48.21 [44.59, 51.83]	0.84 [0.78, 0.90]	1351.38 [1249.83, 1452.93]
Schnabellänge	-5.20 [-14.76, 4.36]	-0.04 [-0.10, 0.03]	-56.89 [-161.46, 47.68]
Geschlecht: männlich	358.63 [276.85, 440.41]	0.45 [0.34, 0.55]	358.63 [276.85, 440.41]
Num.Obs.	333	333	333
R ²	0.807	0.807	0.807

In der Spalte Beta ist auch das Körpergewicht z-standardisiert.

4.1.3 - t-Test, Korrelation und Regression: ein Modell

Im Bachelorstudium lernt man die statistischen Standardverfahren meist als getrennte Werkzeuge, häufig verbunden mit einem Entscheidungsbaum: Zwei Gruppen vergleichen – t-Test. Zwei kontinuierliche Variablen – Korrelation. Mehr als zwei Gruppen – Varianzanalyse. Jacob Cohen (1968) hat schon vor über fünfzig Jahren darauf hingewiesen, dass diese Trennung künstlich ist. Fast alle dieser Verfahren sind Spezialfälle des Allgemeinen Linearen Modells (vgl. auch Lindeløv, 2019).

Der t-Test als Regression. Kodiert man die Gruppenzugehörigkeit als Dummy-Variable mit den Werten 0 und 1 (etwa 0 = weiblich, 1 = männlich), ergibt sich das Modell $y_i = \beta_0 + \beta_1 \cdot \text{männlich}_i + \varepsilon_i$. Für weibliche Tiere ($x = 0$) sagt es β_0 vorher, für männliche ($x = 1$) $\beta_0 + \beta_1$. Der Achsenabschnitt ist also der Mittelwert der Referenzgruppe, und die Steigung β_1 ist exakt die.

Mittelwertsdifferenz. Der t-Wert für β_1 ist identisch mit dem des klassischen t-Tests für unabhängige Stichproben (mit der Annahme gleicher Varianzen). Die Regressionsgerade verbindet einfach die beiden Gruppenmittelwerte.

Die Korrelation als Regression. Standardisiert man x und y vor der Schätzung, ist die Steigung der einfachen Regression exakt der **Korrelationskoeffizient** r . Entsprechend ist der Determinationskoeffizient R^2 der einfachen Regression das Quadrat der Korrelation, und auch hier stimmen die Signifikanztests überein.

Der t-Test als Regression. Kodiert man die Gruppenzugehörigkeit als Dummy-Variable mit den Werten 0 und 1 (etwa 0 = weiblich, 1 = männlich), ergibt sich das Modell $y_i = \beta_0 + \beta_1 \cdot \text{männlich}_i + \varepsilon_i$. Für weibliche Tiere ($x = 0$) sagt es β_0 vorher, für männliche ($x = 1$) $\beta_0 + \beta_1$. Der Achsenabschnitt ist also der Mittelwert der Referenzgruppe, und die Steigung β_1 ist exakt die **Mittelwertsdifferenz**. Der t-Wert für β_1 ist identisch mit dem des klassischen t-Tests für unabhängige Stichproben mit der Annahme gleicher Varianzen. Das lässt sich direkt in R überprüfen:

```
t.test(masse_g ~ geschlecht, data = pinguine, var.equal =
TRUE) |>
  broom::tidy() |>
  select(estimate, statistic, p.value)
```

```
# A tibble: 1 × 3
  estimate statistic p.value
  <dbl>      <dbl>   <dbl>
1   -683.      -8.54 4.90e-16
```

```
lm(masse_g ~ geschlecht, data = pinguine) |>
  broom::tidy() |>
  select(term, estimate, statistic, p.value)
```

```
# A tibble: 2 × 4
  term                estimate statistic p.value
  <chr>              <dbl>      <dbl>   <dbl>
1 (Intercept)       3862.      68.0 1.70e-196
2 geschlechtmännlich  683.        8.54 4.90e- 16
```

Mittelwertsdifferenz, t-Wert und p-Wert stimmen überein. Nur das Vorzeichen unterscheidet sich, weil `t.test()` die erste Gruppe minus die zweite rechnet, `lm()` dagegen die zweite Gruppe minus die Referenzgruppe.

Die Korrelation als Regression. Standardisiert man x und y vor der Schätzung, ist die Steigung der einfachen Regression exakt der **Korrelationskoeffizient** r . Entsprechend ist der Determinationskoeffizient R^2 der einfachen Regression das Quadrat der Korrelation, und auch hier stimmen die Signifikanztests überein:

```
cor.test(~ masse_g + flosse_mm, data = pinguine) |>
  broom::tidy() |>
  select(estimate, statistic, p.value)
```

```
# A tibble: 1 × 3
  estimate statistic p.value
  <dbl>      <dbl>   <dbl>
1   0.873      32.6 3.13e-105
```

```
m_kor <- lm(scale(masse_g) ~ scale(flosse_mm), data =
pinguine)
broom::tidy(m_kor) |>
  select(term, estimate, statistic, p.value)
```

```
# A tibble: 2 × 4
  term                estimate statistic p.value
```

```

      <chr>          <dbl>    <dbl>    <dbl>
1 (Intercept)      -5.32e-16 -1.99e-14 1.000e+ 0
2 scale(flosse_mm)  8.73e- 1  3.26e+ 1 3.13 e-105

```

```

c(r_quadrat
cor(pinguine$masse_g, pinguine$flosse_mm)^2,
R2      = summary(m_kor)$r.squared)

```

```

r_quadrat      R2
0.7620922 0.7620922

```

Die Varianzanalyse als Regression. Hat ein kategorialer Prädiktor mehr als zwei Stufen, etwa die drei Pinguinarten, braucht man mehr als eine Dummy-Variable. Bei drei Arten genügen zwei: eine für Chinstrap und eine für Gentoo, während die Adelie-Pinguine als **Referenzkategorie** bei beiden den Wert 0 haben. Der Achsenabschnitt ist dann der Mittelwert der Referenzkategorie, und die beiden Gewichte sind die Abweichungen der anderen Arten von ihr. Der F-Test der einfaktoriellen Varianzanalyse ist nichts anderes als der Test, ob beide Gewichte gleichzeitig null sind. Die klassische ANOVA-Tabelle aus `aov()` und die Tabelle, die `anova()` für das entsprechende lineare Modell erzeugt, sind deshalb identisch:

```
aov(masse_g ~ art, data = pinguine) |> summary()
```

```

      Df    Sum Sq Mean Sq F value Pr(>F)
art      2 145190219 72595110   341.9 <2e-16 ***
Residuals 330  70069447   212332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

```

```
lm(masse_g ~ art, data = pinguine) |> anova()
```

Analysis of Variance Table

```

Response: masse_g
      Df    Sum Sq Mean Sq F value    Pr(>F)
art      2 145190219 72595110   341.89 < 2.2e-16 ***
Residuals 330  70069447   212332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

```

Nimmt man zusätzlich einen kontinuierlichen Prädiktor auf, wird daraus die Kovarianzanalyse (ANCOVA). Tabelle 4.2 fasst die wichtigsten Entsprechungen zusammen.

Tabelle 4.2: Klassische Testverfahren als Spezialfälle des Allgemeinen Linearen Modells (nach Lindeløv, 2019). D_j steht für eine Dummy-Variable.

Klassisches Verfahren	Als lineares Modell	In R
Einstichproben-t-Test	$y = \beta_0$	<code>lm(y ~ 1)</code>
t-Test (unabhängige Stichproben)	$y = \beta_0 + \beta_1 \cdot \text{Gruppe}$	<code>lm(y ~ gruppe)</code>
t-Test (abhängige Stichproben)	$y_2 - y_1 = \beta_0$	<code>lm(y2 - y1 ~ 1)</code>
Pearson-Korrelation	$z_y = \beta_1 z_x$	<code>lm(scale(y) ~ scale(x))</code>
Einfaktorielle ANOVA	$y = \beta_0 + \beta_1 D_1 + \beta_2 D_2 + \dots$	<code>lm(y ~ faktor)</code>
Mehrfaktorielle ANOVA	zusätzlich Produktterme der Dummies	<code>lm(y ~ f1 * f2)</code>
ANCOVA	$y = \beta_0 + \beta_1 \cdot \text{Gruppe} + \beta_2 x$	<code>lm(y ~ gruppe + x)</code>
Multiple Regression	$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots$	<code>lm(y ~ x1 + x2)</code>

Diese Einsicht ist weit mehr als eine mathematische Kuriosität. Wer das Allgemeine Lineare Modell verstanden hat, muss nicht mehr ein Dutzend Verfahren mit je eigenen Annahmen auswendig lernen, sondern ein einziges Modell mit einem einzigen Satz von Annahmen. Auch wird dadurch nochmals deutlich, dass die Wahl des Analyseverfahrens keinerlei Auswirkungen auf die Interpretation der Ergebnisse im Sinne kausaler Effekte oder nicht-kausaler Zusammenhänge haben kann. Nur weil Sie statt einer Regression eine ANOVA verwenden bedeutet das nicht, dass Sie plötzlich eine kausale Interpretation der Ergebnisse vornehmen können.

Der **Welch-Test**, die Voreinstellung von `t.test()` in R, passt nicht ganz in dieses Schema: Er lässt ungleiche Varianzen zu und entspricht deshalb einem linearen Modell mit gruppenspezifischer Fehlervarianz.

4.2 - Wie die Parameter geschätzt werden

Das Modell in Gleichung (4.2) beschreibt, wie die Daten zustande gekommen sein *könnten*. Die wahren Parameter $\beta_0, \dots, \beta_k$ und σ kennen wir nicht; wir müssen sie aus den Daten schätzen. Dafür gibt es zwei klassische Prinzipien, die im linearen Modell zum selben Ergebnis führen, aber auf sehr unterschiedlichen Ideen beruhen.

4.2.1 - Die Methode der kleinsten Quadrate

Die **Methode der kleinsten Quadrate** (*ordinary least squares*, OLS) wurde vor über 200 Jahren zum ersten Mal von Gauß entwickelt um die Bahnen von Planeten aus imperfekten Beobachtungen besser schätzen zu können. Die Idee ist für den Fall der linearen Regression auch sehr anschaulich. Für jede denkbare Gerade kann man für jeden Datenpunkt ausrechnen, wie weit er vertikal von der Geraden entfernt liegt – das

ist sein Residuum $e_i = y_i - \hat{y}_i$ (Abbildung 4.2). Als beste Gerade gilt die, bei der die Summe der quadrierten Residuen am kleinsten ist:

$$SS_{\text{Res}} = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \rightarrow \min \quad (4.4)$$

Durch das Quadrieren zählen positive und negative Abweichungen gleichermaßen, und große Abweichungen werden stärker gewichtet als kleine. Für das lineare Modell lassen sich die OLS-Schätzer der Modellparameter dadurch sehr schnell ohne ausprobieren aus den Matrizen berechnen: $\hat{\beta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$.

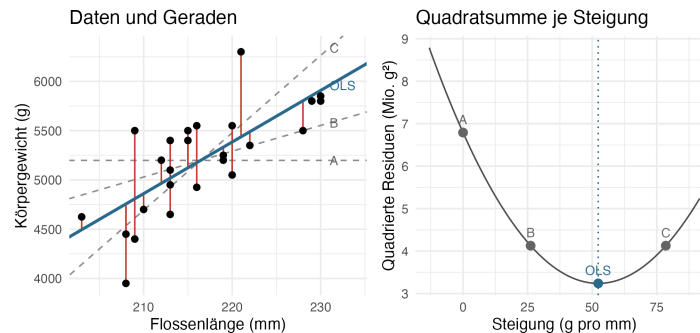


Abbildung 4.2: Das Prinzip der kleinsten Quadrate am Beispiel von 25 zufällig gezogenen Gentoo-Pinguinen. Links: Die schwarzen Punkte sind einzelne Pinguine, die roten Linien die Residuen der Regressionsgeraden (blau). Die gestrichelten Geraden A bis C haben andere Steigungen und passen schlechter; A ist die waagerechte Gerade auf Höhe des Mittelwerts. Alle Geraden verlaufen durch den Schwerpunkt der Daten. Rechts: Summe der quadrierten Residuen in Abhängigkeit von der Steigung. Die Regressionsgerade (OLS) liegt genau im Minimum der Kurve, die alternativen Geraden liegen höher. B und C weichen gleich weit von der optimalen Steigung ab und haben deshalb dieselbe Quadratsumme.

4.2.2 - Maximum-Likelihood-Schätzung

Das zweite Prinzip, die **Maximum-Likelihood-Schätzung** (ML), wurde von Sir Ronald Fisher (1922) ausgearbeitet und ist für den Rest dieses Buches noch wichtiger als die kleinsten Quadrate. Die Idee dreht die übliche Perspektive der Wahrscheinlichkeitsrechnung um. Normalerweise fragen wir: Wie wahrscheinlich sind bestimmte Daten, wenn die Parameter bekannt sind? Bei der ML-Schätzung sind die Daten gegeben, und wir fragen: **Für welche Parameterwerte wären genau diese Daten am plausibelsten?**

Das ALM in Gleichung (4.2) sagt für jeden Fall eine Normalverteilung der y -Werte voraus, deren Mittelwert auf der Regressionsgeraden liegt und deren Streuung σ beträgt (Abbildung 4.3, links). Für einen einzelnen Beobachteten Wert können wir ablesen, wie wahrscheinlich dieser ist, indem wir die Dichte dieser Verteilung an der Stelle des Prädiktor-Wertes ist. Liegt der Punkt nahe an der Geraden, ist die Dichte und damit die Wahrscheinlichkeit hoch; liegt er weit weg, ist die Dichte und damit die Wahrscheinlichkeit niedrig. Weil die Beobachtungen unabhängig sind, ist der **Likelihood** das Produkt der einzelnen Dichten der einzelnen Beobachtungen ($\prod_{i=1}^n$).

Likelihood ist nicht dasselbe wie Wahrscheinlichkeit. Die Likelihood ist eine Funktion der Parameter bei festen Daten und summiert sich über alle Parameterwerte nicht zu eins.

$$L(\beta_0, \dots, \beta_k, \sigma) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - \hat{y}_i)^2}{2\sigma^2}\right) \quad (4.5)$$

Produkte vieler kleiner Zahlen sind rechnerisch unhandlich, deshalb arbeitet man fast immer mit dem Logarithmus, der **Log-Likelihood**. Aus dem Produkt der einzelnen Beobachtungen wird eine Summe der quadrierten Differenzen:

$$\ell = \ln L = -\frac{n}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (4.6)$$

Ein Blick auf die Formel Gleichung (4.6) zeigt, warum im linearen Modell der OLS-Schätzer auch dem likelihood Prinzip entspricht: Der zweite Term entspricht ja gerade der Summe der quadrierten Abweichungen zwischen den beobachteten y_i und den vorhergesagten $\hat{y}_i$ Werten. Die Regressionsgewichte stecken nur im zweiten Term, und der ist bis auf einen negativen Faktor genau die Summe der quadrierten Residuen. Die Log-Likelihood wird also genau dann maximal, wenn SS_{Res} minimal ist. Die ML-Schätzer der Regressionsgewichte sind mit den OLS-Schätzern identisch (Abbildung 4.3, rechts).

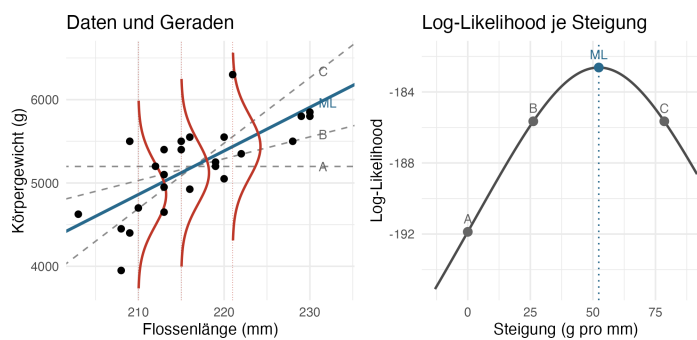


Abbildung 4.3: Maximum Likelihood für dieselben 25 Gentoo-Pinguine wie in Abbildung 4.2. Links: Das lineare Modell unterstellt an jeder Stelle von x eine Normalverteilung um die Regressionsgerade (blau). Die gestrichelten Geraden A bis C sind dieselben Alternativen wie in Abbildung 4.2; unter ihnen liegen die beobachteten Werte weiter von den Zentren der Normalverteilungen entfernt und sind deshalb weniger wahrscheinlich. Rechts: Profil-Log-Likelihood für verschiedene Werte der Steigung (Achsenabschnitt und σ jeweils optimal gewählt). Das Maximum liegt exakt beim OLS-Schätzer (ML, gepunktete Linie), die alternativen Geraden erreichen eine geringere Log-Likelihood.

Wenn beide Verfahren dasselbe liefern, warum dann der Aufwand? Weil die kleinsten Quadrate eine Sackgasse sind, sobald wir das lineare Modell verlassen. Für eine logistische Regressionen, Mehrebenenmodelle, oder Faktorenanalysen gibt es keine sinnvolle Summe quadrierter Residuen, die man minimieren könnte. Eine Likelihood lässt sich dagegen für praktisch jedes statistische Modell aufschreiben, und die Logik bleibt immer dieselbe: Man sucht die Parameter, unter denen die beobachteten Daten am plausibelsten sind. Meist gibt es dafür keine geschlossene Lösung mehr, sondern es muss ausprobiert werden. Das macht man nicht wahllos, sondern für dieses ausprobieren gibt es unterschiedliche Verfahren zur sogenannten (*iterative Optimierung*) die mal schneller mal langsamer diese optimalen Werte finden.

4.3 - Varianzaufklärung

4.3.1 - Der Determinationskoeffizient R^2

Auch das Modell mit den optimalen Koeffizienten ist anders als Sie selbst leider oft nicht gut gennuug.

Nachdem man optimale Schätzer für die Parameter des Modells identifiziert hat, stellt sich die Frage wie gut das Modell die Daten beschreibt? Nur weil es das Beste ist, heißt das nicht, dass es gut ist. Die gebräuchlichste Antwort ist der **Determinationskoeffizient R^2** . Er beruht auf einer einfachen Zerlegung: Die Gesamtvariation der abhängigen Variable (SS_{Total} , d.h. die quadrierten Abweichungen vom Mittelwert) teilt sich auf in einen Teil, den das Modell vorhersagt (SS_{Modell}), und einen Teil, der als Residuum übrig bleibt (SS_{Res}):

$$R^2 = \frac{SS_{\text{Modell}}}{SS_{\text{Total}}} = 1 - \frac{SS_{\text{Res}}}{SS_{\text{Total}}} \quad (4.7)$$

R^2 ist also der Anteil der Varianz in y , der durch die Prädiktoren „erklärt“ wird, und liegt zwischen 0 und 1. Die Wurzel R heißt **multiple Korrelation**; sie ist die Korrelation zwischen beobachteten und vorhergesagten Werten.

„Erklärt“, ist hier rein statistisch gemeint: Ein Teil der Varianz lässt sich *vorhersagen*. Über kausale Mechanismen oder Ursachen sagt R^2 nichts.

Wie groß ist ein großes R^2 ? Wer von den Pinguinen an hohe Werte gewöhnt ist – Flossenlänge und Körpergewicht korrelieren zu $r = .87$, die Flossenlänge allein erklärt also 76 % der Varianz des Gewichts –, wird von den Zusammenhängen in der Praxis überrascht (Tabelle 4.3).

Tabelle 4.3: Korrelationen und Varianzaufklärung für einige bekannte Zusammenhänge (nach Meyer et al., 2001; Orben & Przybylski, 2019) und für den Pinguin-Datensatz.

Zusammenhang	r	R^2
Aspirin und geringeres Herzinfarktrisiko	.02	0,04 %
Rauchen und Lungenkrebs innerhalb von 25 Jahren	.08	0,6 %
Nutzung digitaler Medien und Wohlbefinden Jugendlicher	$\approx .06$	$\leq 0,4$ %
Psychotherapie und späteres Wohlbefinden	.32	10 %
Größe und Gewicht (Erwachsene in den USA)	.44	19 %
Flossenlänge und Körpergewicht (Pinguine)	.87	76 %

R^2 hat eine unangenehme Eigenschaft: Es kann durch zusätzliche Prädiktoren nur steigen, nie sinken, selbst wenn diese reines Rauschen sind. Jeder zusätzliche Prädiktor passt sich ein kleines bisschen an die Zufälligkeiten der Stichprobe an. Das **korrigierte** R^2_{adj} bestraft deshalb jeden zusätzlichen Prädiktor und schätzt die Varianzaufklärung in der Population besser.

4.3.2 - Inkrementelle Varianzaufklärung

In der angewandten Psychologie interessiert es selten, wie viel Varianz ein Modell *insgesamt* erklärt. Viel häufiger lautet die Frage: Bringt ein neuer Prädiktor etwas *über das hinaus*, was wir schon wissen? Lohnt sich ein aufwändiges Assessment-Center, wenn wir bereits einen Intelligenztest einsetzen? Sagt ein neuer Fragebogen zur Resilienz Burnout über die bekannten Big-Five-Faktoren hinaus vorher? Diese Frage nach der **inkrementellen Validität** (Hunsley & Meyer, 2003) beantwortet man mit einer **hierarchischen Regression**: Man schätzt nacheinander mehrere geschachtelte Modelle, in die die Prädiktoren blockweise in einer theoretisch begründeten Reihenfolge aufgenommen werden, und betrachtet den Zuwachs

$$\Delta R^2 = R^2_{\text{komplex}} - R^2_{\text{einfach}} \quad (4.8)$$

Ob dieser Zuwachs statistisch bedeutsam ist, kann man anhand eines F-Tests prüfen, bei dem k die Anzahl der Prädiktoren im jeweiligen Modell ist:

$$F = \frac{\Delta R^2 / (k_{\text{komplex}} - k_{\text{einfach}})}{(1 - R^2_{\text{komplex}}) / (n - k_{\text{komplex}} - 1)} \quad (4.9)$$

Ein klassisches Beispiel aus der Wirtschaftspsychologie ist die Meta-Analyse von Schmidt und Hunter (1998) zur Personalauswahl. Allgemeine kognitive Fähigkeiten sagten dort Berufserfolg mit einer Validität von $r = .51$ vorher. Nahm man einen Integritätstest hinzu, stieg die multiple Korrelation auf $R = .65$, was einem Zuwachs in der Varianzaufklärung von $\Delta R^2 = .65^2 - .51^2 \approx .16$ entspricht. Kaum ein anderer Befund hat die Praxis der Personalauswahl so geprägt. Er illustriert allerdings auch, dass solche Zahlen nicht in Stein gemeißelt sind: Eine Neubewertung der zugrunde liegenden Korrekturen für Varianzeinschränkung kam zu deutlich niedrigeren Validitäten und einer veränderten Rangfolge der Verfahren (Sackett et al., 2022).

Für einen *einzelnen* zusätzlichen Prädiktor ist ΔR^2 genau die quadrierte **semipartielle Korrelation** sr^2 : der Anteil der Gesamtvarianz von y , den nur dieser Prädiktor erklärt.

4.3.3 - Wie man ΔR^2 (nicht) interpretiert

So einfach die Berechnung ist, so leicht lässt sich ΔR^2 falsch interpretieren. Vier Punkte sollten Sie im Blick behalten.

Die Reihenfolge entscheidet. Korrelieren die Prädiktoren untereinander, teilen sie sich einen Teil der Varianz, die sie an y aufklären. Man kann sich das wie in Abbildung 4.4 vorstellen: Der Bereich b , in dem sich alle drei Kreise überlappen, wird in der hierarchischen Regression immer demjenigen Prädiktor zugeschlagen, der *zuerst* ins Modell kommt. Nimmt man X_1 zuerst auf, erklärt es $a + b$, und X_2 bringt nur noch den Zuwachs c . In umgekehrter Reihenfolge ist es genau andersherum. Derselbe Prädiktor kann also je nach Reihenfolge viel oder wenig inkrementelle Varianz aufklären. Die Reihenfolge muss deshalb vorab theoretisch begründet sein – etwa: Was ist das etablierte Verfahren, was das neue, zu prüfende? Die Frage „Welcher Prädiktor

ist wichtiger?, lässt sich mit ΔR^2 allein nicht beantworten; dafür gibt es eigene Verfahren wie die Dominanzanalyse oder relative Gewichte (Johnson & LeBreton, 2004).

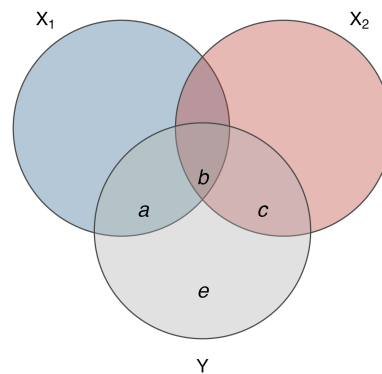


Abbildung 4.4: Varianzzerlegung bei zwei korrelierten Prädiktoren (Venn-Diagramm). Die Fläche von Y steht für seine Varianz. X_1 allein erklärt $a + b$, X_2 allein $b + c$; beide zusammen $a + b + c$. Der gemeinsame Anteil b wird in der hierarchischen Regression dem zuerst aufgenommenen Prädiktor zugerechnet. e ist die unerklärte Varianz.

Signifikanz ist nicht gleich Relevanz. Wie immer gilt auch bei Betrachtungen der inkrementellen Varianz, dass Signifikanz alleine noch kein Garant dafür ist, dass der Zuwachs auch praktisch relevant ist. Bei großen Stichproben praktisch jeder noch so kleine Zuwachs signifikant. Berichten Sie deshalb immer ΔR^2 (am besten mit Konfidenzintervall) und nicht nur den p-Wert des F-Tests.

Messfehler täuschen inkrementelle Validität vor. Eine weitere wichtige Einschränkung betrifft die Rolle der Reliabilität des ersten Prädiktors. Wer zeigen will, dass ein neues Konstrukt X_2 „über“, ein bekanntes Konstrukt X_1 hinaus etwas vorhersagt, kontrolliert eigentlich nicht das Konstrukt, sondern nur dessen imperfekte *Messung*. Ist diese messfehlerbehaftet, bleibt ein Teil der Varianz von X_1 unkontrolliert, und diesen Rest kann X_2 aufsammeln, sofern es mit X_1 korreliert. Westfall und Yarkoni (2016) haben gezeigt, dass dadurch in realistischen Szenarien massenhaft scheinbare inkrementelle Effekte entstehen. Eine kleine Simulation im Stil der Einleitung macht das Problem greifbar (Abbildung 4.5): Wir erzeugen Daten, in denen y *ausschließlich* vom Konstrukt T abhängt. X_1 ist eine mehr oder weniger reliable Messung von T , und X_2 misst ebenfalls T und korreliert mit diesem zu .60, hat aber selbst keinerlei Einfluss auf y . In diesem Szenario sollte X_2 also keinerlei inkrementelle Varianzaufklärung gegenüber X_1 besitzen. Wir schauen aber einfach mal was passiert indem wir 2.000 Stichproben generieren und checken, wie oft X_2 in der Regression $y \sim X_1 + X_2$ signifikant wird.

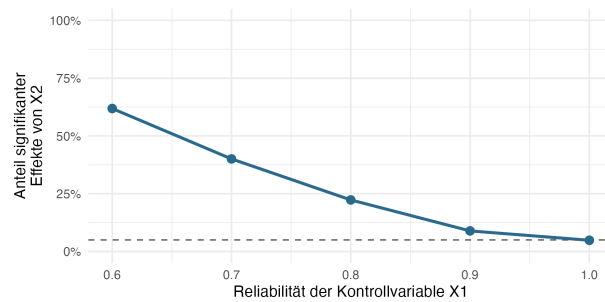


Abbildung 4.5: Anteil der Stichproben ($n = 300$, je 2.000 Wiederholungen), in denen ein Prädiktor ohne eigenen Effekt signifikant „inkrementelle Validität“, zeigt, in Abhängigkeit von der Reliabilität der Kontrollvariable. Die gestrichelte Linie markiert die nominelle Fehler-rate von 5 %.

Nur bei perfekter Messung hält der Test sein Fehlerniveau von 5 % ein. Schon bei einer Reliabilität von .90, die für psychologische Skalen als sehr gut gilt, findet man den nicht existierenden Effekt weit häufiger als erwartet, und bei typischen Reliabilitäten von .70 bis .80 findet man in etwa einem viertel der Fälle eine schein- inkrementelle Varianz. Die Lehre daraus: Behauptungen über inkrementelle Validität *auf Konstruktebene* sind mit einer einfachen hierarchischen Regression nicht zu belegen. Wer sie ernsthaft prüfen will, braucht Modelle, die den Messfehler explizit berücksichtigen aka Strukturgleichungsmodelle.

4.4 - Annahmen des linearen Modells

Jedes statistische Modell beruht auf Annahmen, und Anscombes Quartett aus dem Kapitel über Abbildungen hat gezeigt, dass man ihre Verletzung den üblichen Kennwerten nicht ansieht. Die Annahmen des linearen Modells werden oft als lose Liste auswendig gelernt. Gelman, Hill und Vehtari (2020) ordnen sie dagegen nach ihrer **Wichtigkeit**, und diese Reihenfolge ist aufschlussreich, weil sie ziemlich genau umgekehrt zu der Aufmerksamkeit verläuft, die den einzelnen Annahmen in der Praxis gewidmet wird.

1. **Validität.** Die Daten müssen zur Forschungsfrage passen: Misst das Outcome das, was uns eigentlich interessiert? Sind alle relevanten Prädiktoren im Modell? Kein statistischer Trick der Welt die falsche Frage beantwortet.
2. **Repräsentativität.** Die Stichprobe muss die Population abbilden, über die wir Aussagen machen wollen.
3. **Linearität und Additivität.** Der Erwartungswert von y ist eine lineare Funktion der Prädiktoren, und ihre Effekte addieren sich. Ist der Zusammenhang gekrümmt (wie im zweiten Datensatz von Anscombe) oder hängt der Effekt eines Prädiktors von einem anderen ab, ist das Modell falsch spezifiziert. Abhilfe schaffen Transformationen, quadratische Terme oder Interaktionen.
4. **Unabhängigkeit der Fehler.** Die Residuen verschiedener Beobachtungen dürfen nicht zusammenhängen. Diese Annahme ist verletzt, wenn Personen mehrfach gemessen werden oder in Gruppen (Teams, Schulklassen, Kliniken) organisiert sind. Die Standardfehler werden dann meist dramatisch unterschätzt. Dafür gibt es Mehrebenenmodelle, denen wir ein eigenes Kapitel widmen.

„Far better an approximate answer to the right question, which is often vague, than an exact answer to the wrong question, which can always be made precise“ — (Tukey, 1962)

5. **Homoskedastizität.** Die Fehlervarianz σ^2 ist für alle Werte der Prädiktoren gleich. Ist sie es nicht (**Heteroskedastizität**), bleiben die Gewichte zwar unverzerrt, aber die Standardfehler und damit p-Werte und Konfidenzintervalle stimmen nicht mehr. Ein einfaches Gegenmittel sind **heteroskedastizitätskonsistente (robuste) Standardfehler**, etwa vom Typ HC3 (Long & Ervin, 2000).
6. **Normalverteilung der Fehler.** Sie ist für die Schätzung der Gewichte gar nicht nötig und für Tests und Konfidenzintervalle bei nicht allzu kleinen Stichproben dank des zentralen Grenzwertsatzes ziemlich unkritisch. Wichtig wird sie vor allem, wenn man einzelne Werte vorhersagen will (Vorhersageintervalle).

! Normalverteilt sein müssen die Residuen, nicht die Variablen

Ein hartnäckiges Missverständnis lautet, dass die abhängige Variable oder gar die Prädiktoren normalverteilt sein müssen (Williams et al., 2013). Das Modell macht über die Verteilung der Prädiktoren überhaupt keine Annahmen, und die abhängige Variable ist schon allein wegen der Unterschiede zwischen den Gruppen oder entlang der Prädiktoren oft nicht normalverteilt. Das Körpergewicht der Pinguine ist z. B. zweigipflig, weil die Gentoo-Pinguine so viel schwerer sind. Ein Kolmogorov-Smirnov-Test auf die Rohwerte von y prüft also schlicht die falsche Frage.

Das wichtigste Werkzeug zur Prüfung der Annahmen ist, Sie ahnen es, eine Abbildung: das Streudiagramm der **Residuen gegen die vorhergesagten Werte**. Wenn alles in Ordnung ist, sieht man darin eine strukturlose, gleichmäßig breite Punktwolke um die Nulllinie. Jede Struktur ist ein Hinweis auf eine verletzte Annahme (Abbildung 4.6).

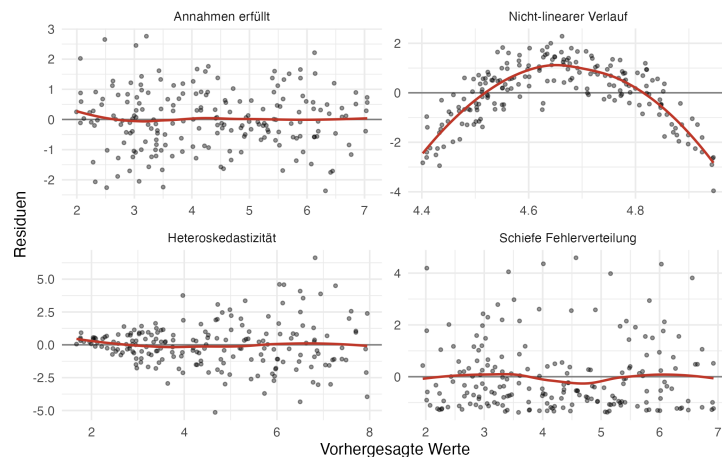


Abbildung 4.6: Residuen gegen vorhergesagte Werte für vier simulierte Datensätze (je $n = 200$). Eine Glättungskurve (rot) hilft, systematische Muster zu erkennen.

Ein gebogenes Muster deutet auf einen nicht-linearen Zusammenhang hin, eine sich trichterförmig öffnende Punktwolke auf Heteroskedastizität, und eine asymmetrische Verteilung um die Nulllinie mit einzelnen weit oben liegenden Punkten auf eine schiefe Fehlerverteilung. Formale Tests für einzelne Annahmen (etwa der Breusch-Pagan-Test auf Heteroskedastizität) gibt es zwar, sie sind aber mit Vorsicht zu genießen: In

kleinen Stichproben übersehen sie relevante Verletzungen, in großen schlagen sie bei völlig harmlosen Abweichungen an. Der Blick auf die Abbildung ist meistens aufschlussreicher.

Neben diesen Annahmen gibt es zwei Eigenschaften der Daten, die man prüfen sollte, obwohl sie streng genommen keine Modellannahmen sind.

1. **Multikollinearität** liegt vor, wenn Prädiktoren sehr hoch untereinander korrelieren. Die Schätzungen bleiben dann unverzerrt, werden aber sehr unpräzise, weil das Modell kaum unterscheiden kann, welchem Prädiktor es die gemeinsame Varianz zuordnen soll. Der **Varianzinflationsfaktor** (VIF) gibt an, um welchen Faktor die Varianz eines Gewichts dadurch aufgebläht wird; Werte über 5 oder 10 gelten als Warnsignal.
2. **Einflussreiche Fälle** sind einzelne Beobachtungen, die das Ergebnis stark verändern würden, wenn man sie wegließe – wie der Ausreißer im vierten Datensatz von Anscombe. Das gebräuchlichste Maß dafür ist **Cooks Distanz**. Und wie im Kapitel zum Datenmanagement gilt: Einflussreiche Fälle werden nicht stillschweigend gelöscht, sondern berichtet, und das Ergebnis wird mit und ohne sie gezeigt.

4.5 - Regression mit R

Nach so viel Konzept geht es nun an die praktische Umsetzung. Sie werden sehen, dass die Einheitlichkeit des Allgemeinen Linearen Modells sich direkt in R widerspiegelt: Fast alles, was wir in diesem Kapitel besprochen haben, lässt sich mit einer einzigen Funktion, `lm()`, und einer einheitlichen Formelschreibweise erledigen.

4.5.1 - Beispieldatensatz

Wir arbeiten weiter mit den Palmer-Pinguinen. Im Kapitel zum Datenmanagement haben wir aus den Rohdaten einen analysefertigen Datensatz gebaut; der Kürze halber beginnen wir hier mit der bereits aufgeräumten Version `penguins` und bringen sie mit den bekannten Verben in die Form, die wir brauchen. Weil alle Modelle, die wir vergleichen wollen, auf *denselben* Fällen beruhen müssen, schließen wir Tiere mit fehlenden Werten vorab aus.

```
library(tidyverse)
library(palmerpenguins)

pinguine <- penguins |>
  select(art = species, insel = island,
         schnabel_mm = bill_length_mm, schnabeltiefe_mm =
           = bill_depth_mm,
         flosse_mm = flipper_length_mm, masse_g =
           body_mass_g,
         geschlecht = sex) |>
  drop_na() |>
  mutate(geschlecht = fct_recode(geschlecht,
                                weiblich = "female",
                                männlich = "male"))

glimpse(pinguine)
```

```

Rows: 333
Columns: 7
$ art           <fct> Adelie, Adelie, Adelie, Adelie,
Adelie, Adelie, Adeli...
$ insel         <fct> Torgersen, Torgersen, Torgersen,
Torgersen, Torgersen...
$ schnabel_mm   <dbl> 39.1, 39.5, 40.3, 36.7, 39.3,
38.9, 39.2, 41.1, 38.6,...
$ schnabeltiefe_mm <dbl> 18.7, 17.4, 18.0, 19.3, 20.6,
17.8, 19.6, 17.6, 21.2,...
$ flosse_mm     <int> 181, 186, 195, 193, 190, 181,
195, 182, 191, 198, 185...
$ masse_g       <int> 3750, 3800, 3250, 3450, 3650,
3625, 4675, 3200, 3800,...
$ geschlecht    <fct> männlich, weiblich, weiblich,
weiblich, männlich, wei...

```

`lm()` schließt Fälle mit fehlenden Werten stillschweigend aus. Das kann immer dann zu Problemen führen, wenn man Modelle mit unterschiedlichen Prädiktoren vergleicht. Da man dann ggf. nicht nur die Modelle sondern vor allem die Stichproben miteinander vergleicht.

4.5.2 - Die Funktion `lm()` und die Formelschreibweise

Lineare Modelle schätzt man in R mit `lm()` (für *linear model*). Die Funktion braucht zwei Dinge: eine **Formel**, die das Modell beschreibt, und den Datensatz. Links von der Tilde `~` steht die abhängige Variable, rechts stehen die Prädiktoren. Den Achsenabschnitt müssen Sie nicht angeben, er wird automatisch geschätzt. Wir beginnen mit der einfachen Regression des Körpergewichts auf die Flossenlänge:

```

m_flosse <- lm(masse_g ~ flosse_mm, data = pinguine)
m_flosse

```

```

Call:
lm(formula = masse_g ~ flosse_mm, data = pinguine)

Coefficients:
(Intercept)    flosse_mm
   -5872.09         50.15

```

Die knappe Ausgabe zeigt nur die geschätzten Koeffizienten. Pro Millimeter Flossenlänge sind die Tiere im Mittel um etwa 50 g schwerer. Der Achsenabschnitt von -5872 g ist das vorhergesagte Gewicht eines Pinguins mit einer Flossenlänge von 0 mm – ein Wert, der biologisch keinen Sinn ergibt und weit außerhalb der Daten liegt. Das ist bei Achsenabschnitten häufig und kein Grund zur Sorge. Wer einen interpretierbaren Achsenabschnitt möchte, **zentriert** den Prädiktor vorher (zieht also seinen Mittelwert ab); dann ist der Achsenabschnitt das vorhergesagte Gewicht bei durchschnittlicher Flossenlänge.

Eine ausführlichere Ausgabe mit Standardfehlern, t-Werten, p-Werten und R^2 erhalten Sie mit `summary()`:

```
summary(m_flosse)
```

```

Call:
lm(formula = masse_g ~ flosse_mm, data = pinguine)

```

```

Residuals:
    Min       1Q   Median       3Q      Max
-1057.33 -259.79  -12.24   242.97 1293.89

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -5872.09      310.29  -18.93  <2e-16 ***
flosse_mm     50.15        1.54   32.56  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 393.3 on 331 degrees of freedom
Multiple R-squared:  0.7621,    Adjusted R-squared:  0.7614
F-statistic: 1060 on 1 and 331 DF,  p-value: < 2.2e-16

```

Die Ausgabe ist etwas überladen, aber systematisch aufgebaut. Oben steht zur Erinnerung das Modell, darunter eine Kurzbeschreibung der Residuen, dann die Koeffiziententabelle und unten die Kennwerte des Gesamtmodells: die Standardabweichung der Residuen (*Residual standard error*, $\hat{\sigma}$), R^2 und korrigiertes R^2 sowie der F-Test des Gesamtmodells. Konfidenzintervalle für die Koeffizienten, die im Sinne der *New Statistics* eigentlich wichtiger sind als die p-Werte, liefert `confint()`:

```
confint(m_flosse)
```

```

              2.5 %      97.5 %
(Intercept) -6482.47224 -5261.71313
flosse_mm     47.12339    53.18314

```

Die Formelschreibweise ist deutlich mächtiger, als dieses erste Beispiel vermuten lässt. Tabelle 4.4 zeigt die wichtigsten Bausteine.

Tabelle 4.4: Die wichtigsten Bausteine der Formelschreibweise in R.

Formel	Bedeutung
$y \sim x$	einfache Regression mit Achsenabschnitt
$y \sim x_1 + x_2$	multiple Regression, Effekte addieren sich
$y \sim x_1 * x_2$	Haupteffekte <i>und</i> Interaktion (Kurzform für $x_1 + x_2 + x_1:x_2$)
$y \sim x_1:x_2$	nur der Interaktionsterm
$y \sim 1$	nur der Achsenabschnitt (Nullmodell)
$y \sim 0 + x$ oder $y \sim x - 1$	ohne Achsenabschnitt
$y \sim x + I(x^2)$	quadratischer Term; <code>I()</code> schützt die Rechnung vor der Formelsyntax
$y \sim \log(x)$	transformierter Prädiktor
$y \sim .$	alle übrigen Variablen des Datensatzes als Prädiktoren

Das Objekt, das `lm()` zurückgibt, enthält alles, was man braucht. `coef()`, `confint()`, `fitted()`, `resid()`, `predict()` und `sigma()` ziehen die einzelnen Bestandteile heraus.

4.5.3 - t-Test, Korrelation und ANOVA mit lm()

Die im konzeptuellen Teil behauptete Einheit der Verfahren lässt sich in R direkt nachprüfen. Zuerst der **t-Test**: Wir vergleichen das Gewicht männlicher und weiblicher Tiere einmal mit `t.test()` (mit der Annahme gleicher Varianzen) und einmal mit `lm()`. Für die kompakte Darstellung der Koeffizienten nutzen wir `tidy()` aus dem Paket `broom`, das mit dem `tidyverse` installiert wird:

```
t.test(masse_g ~ geschlecht, data = pinguine, var.equal = TRUE)
```

Two Sample t-test

```
data: masse_g by geschlecht
t = -8.5417, df = 331, p-value = 4.897e-16
alternative hypothesis: true difference in means between
group weiblich and group männlich is not equal to 0
95 percent confidence interval:
 -840.8014 -526.0222
sample estimates:
mean in group weiblich mean in group männlich
      3862.273           4545.685
```

```
lm(masse_g ~ geschlecht, data = pinguine) |>
  broom::tidy()
```

```
# A tibble: 2 × 5
  term          estimate std.error statistic  p.value
<chr>          <dbl>    <dbl>    <dbl>    <dbl>
1 (Intercept)    3862.      56.8      68.0 1.70e-196
2 geschlechtmännlich  683.      80.0       8.54 4.90e-16
```

Der Achsenabschnitt ist der Mittelwert der weiblichen Tiere, das Gewicht `geschlechtmännlich` die Mittelwertsdifferenz, und t-Wert und p-Wert sind identisch. Nur das Vorzeichen des t-Werts unterscheidet sich, weil `t.test()` die erste Gruppe minus die zweite rechnet, `lm()` dagegen die zweite minus die Referenzgruppe.

Die **Korrelation** entspricht der Steigung einer Regression standardisierter Variablen. Mit `scale()` lassen sich die Variablen direkt in der Formel standardisieren:

```
cor.test(~ masse_g + flosse_mm, data = pinguine)
```

Pearson's product-moment correlation

```
data: masse_g and flosse_mm
t = 32.562, df = 331, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8447622 0.8963550
sample estimates:
cor
0.8729789
```

Kategoriale Prädiktoren wandelt R automatisch in Dummy-Variablen um. Die **erste Faktorstufe** ist die Referenzkategorie. Mit `fct_relevel(art, "Gentoo")` machen Sie eine andere Stufe zur Referenz.

```
lm(scale(masse_g) ~ scale(flosse_mm), data = pinguine) |>
  broom::tidy()
```

```
# A tibble: 2 × 5
  term          estimate std.error statistic    p.value
<chr>          <dbl>    <dbl>    <dbl>    <dbl>
1 (Intercept) -5.32e-16    0.0268 -1.99e-14 1.000e+ 0
2 scale(flosse_mm) 8.73e- 1    0.0268  3.26e+ 1 3.13 e-105
```

Und schließlich die **einfaktorielle Varianzanalyse**. Die klassische ANOVA-Tabelle erhält man aus `aov()`; wendet man `anova()` auf das entsprechende lineare Modell an, entsteht exakt dieselbe Tabelle:

```
aov(masse_g ~ art, data = pinguine) |> summary()
```

```
          Df    Sum Sq Mean Sq F value Pr(>F)
art         2 145190219 72595110   341.9 <2e-16 ***
Residuals  330  70069447  212332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1
```

```
lm(masse_g ~ art, data = pinguine) |> anova()
```

```
Analysis of Variance Table

Response: masse_g
          Df    Sum Sq Mean Sq F value    Pr(>F)
art         2 145190219 72595110   341.89 < 2.2e-16 ***
Residuals  330  70069447  212332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1
```

Der Vorteil der `lm()`-Variante zeigt sich, wenn man sich die Koeffizienten ansieht: Sie liefern nicht nur die Information, *dass* sich die Arten unterscheiden, sondern auch, *wie stark* sich Chinstrap- und Gentoo-Pinguine von der Referenzkategorie Adelie unterscheiden.

```
lm(masse_g ~ art, data = pinguine) |>
  broom::tidy(conf.int = TRUE)
```

```
# A tibble: 3 × 7
  term          estimate std.error statistic    p.value
conf.low conf.high
<chr>          <dbl>    <dbl>    <dbl>    <dbl>
<dbl>    <dbl>
1 (Intercept)  3706.      38.1     97.2 6.88e-245
3631.    3781.
2 artChinstrap  26.9      67.7     0.398 6.91e- 1
-106.    160.
3 artGentoo    1386.     56.9    24.4 1.01e- 75
1274.    1498.
```

4.5.4 - Multiple Regression

Nun nehmen wir mehrere Prädiktoren gleichzeitig ins Modell auf. Wir sagen das Körpergewicht aus Flossenlänge, Geschlecht und Art vorher:

```
m_voll <- lm(masse_g ~ flosse_mm + geschlecht + art, data
= pinguine)
summary(m_voll)
```

```
Call:
lm(formula = masse_g ~ flosse_mm + geschlecht + art, data
= pinguine)

Residuals:
    Min       1Q   Median       3Q      Max
-721.8 -195.7   -5.9   198.6   873.7

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   -365.817    532.050  -0.688   0.4922
flosse_mm       20.025      2.846   7.037 1.15e-11 ***
geschlechtmännlich 530.381    37.810  14.027 < 2e-16 ***
artChinstrap   -87.634     46.347  -1.891   0.0595 .
artGentoo       836.260     85.185   9.817 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

Residual standard error: 295.6 on 328 degrees of freedom
Multiple R-squared:  0.8669,    Adjusted R-squared:  0.8653
F-statistic:  534 on 4 and 328 DF,  p-value: < 2.2e-16
```

Die Interpretation folgt der Logik der partiellen Gewichte. Pro Millimeter Flossenlänge sind die Tiere *bei gleichem Geschlecht und gleicher Art* um 20 g schwerer — deutlich weniger als die 50 g aus der einfachen Regression, weil ein Teil des einfachen Zusammenhangs darauf zurückging, dass die schwereren Gentoo-Pinguine auch längere Flossen haben. Männliche Tiere sind bei gleicher Flossenlänge und Art um 530 g schwerer als weibliche. Die Artkoeffizienten beschreiben die Unterschiede zu Adelie-Pinguinen gleichen Geschlechts und gleicher Flossenlänge.

Das Modell unterstellt, dass die Steigung der Flossenlänge für alle Arten und beide Geschlechter gleich ist, dass also die Regressionsgeraden parallel verlaufen. Ob das stimmt, prüft man mit einem **Interaktionsterm**, etwa `flosse_mm * art`. Interaktionen und ihre Interpretation (Moderation) vertiefen wir im nächsten Kapitel zu Varianzanalysen.

Um die Stärke der Prädiktoren trotz unterschiedlicher Einheiten zu vergleichen, kann man **standardisierte Gewichte** berechnen. Am einfachsten geht das, indem man die kontinuierlichen Variablen vorher mit `scale()` z-standardisiert; `as.numeric()` sorgt dafür, dass aus dem Ergebnis wieder ein einfacher Vektor wird. Kategoriale Prädiktoren lassen wir dabei unverändert:

```
pinguine_z <- pinguine |>
  mutate(across(c(masse_g, flosse_mm, schnabel_mm), \(x)
as.numeric(scale(x))))

lm(masse_g ~ flosse_mm + schnabel_mm, data = pinguine_z)
```

```
|>
  broom::tidy()
```

```
# A tibble: 3 × 5
  term          estimate std.error statistic    p.value
<chr>         <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept) -4.02e-16    0.0268 -1.50e-14 1.000e+ 0
2 flosse_mm    8.51e- 1    0.0354  2.40e+ 1 1.74 e-74
3 schnabel_mm  3.37e- 2    0.0354  9.51e- 1 3.42 e- 1
```

4.5.5 - Hierarchische Regression und inkrementelle Varianzaufklärung

Für eine hierarchische Regression schätzt man die geschachtelten Modelle nacheinander. Wir nehmen zuerst die Flossenlänge auf, dann das Geschlecht und zuletzt die Art. Die Modelle speichern wir in einer benannten Liste, weil wir sie später gemeinsam in eine Tabelle übergeben wollen:

```
m1 <- lm(masse_g ~ flosse_mm, data = pinguine)
m2 <- lm(masse_g ~ flosse_mm + geschlecht, data = pinguine)
m3 <- lm(masse_g ~ flosse_mm + geschlecht + art, data = pinguine)

modelle <- list("Modell 1" = m1, "Modell 2" = m2, "Modell 3" = m3)
```

Die F-Tests für die Zuwächse liefert `anova()`, wenn man ihr mehrere geschachtelte Modelle übergibt. Jede Zeile vergleicht ein Modell mit dem vorhergehenden:

```
anova(m1, m2, m3)
```

```
Analysis of Variance Table

Model 1: masse_g ~ flosse_mm
Model 2: masse_g ~ flosse_mm + geschlecht
Model 3: masse_g ~ flosse_mm + geschlecht + art
  Res.Df    RSS Df Sum of Sq    F    Pr(>F)
1     331 51211963
2     330 41795374   1   9416589 107.793 < 2.2e-16 ***
3     328 28653568   2  13141806  75.218 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Die Spalte `RSS` enthält die Residuenquadratsummen, `Df` die Zahl der zusätzlichen Parameter, und `F` und `Pr(>F)` den Test auf den Zuwachs. R^2 und ΔR^2 zeigt `anova()` nicht direkt an; wir berechnen sie mit `map_dbl()`, das eine Funktion auf jedes Element der Liste anwendet. `lag()` holt den Wert der vorherigen Zeile, sodass die Differenz den Zuwachs ergibt. Zusätzlich berechnen wir Cohens f^2 :

```
r2_tabelle <- tibble(
  modell = names(modelle),
  r2      = map_dbl(modelle, \(m) summary(m)$r.squared)
) |>
```

`anova()` mit **einem** Modell liefert Tests für die einzelnen Terme, mit **mehreren** Modellen den Vergleich der Modelle.

```
mutate(
  delta_r2 = r2 - lag(r2, default = 0),
  f2       = delta_r2 / (1 - r2)
)

r2_tabelle
```

```
# A tibble: 3 × 4
  modell      r2 delta_r2    f2
  <chr>    <dbl>   <dbl> <dbl>
1 Modell 1 0.762   0.762  3.20
2 Modell 2 0.806   0.0437 0.225
3 Modell 3 0.867   0.0611 0.459
```

Die Flossenlänge allein erklärt bereits 76 % der Varianz des Körpergewichts. Das Geschlecht bringt weitere 4.4 Prozentpunkte, die Art noch einmal 6.1 Prozentpunkte. (Für das erste Modell ist f^2 in dieser Tabelle nicht als Zuwachs, sondern als Effekt des Gesamtmodells zu lesen.)

Wie stark die inkrementelle Varianzaufklärung von der **Reihenfolge** abhängt, sehen wir, wenn wir die Blöcke umdrehen und die Art zuerst aufnehmen:

```
r2 <- \ (m) summary(m)$r.squared

m_art      <- lm(masse_g ~ art, data = pinguine)
m_art_flosse <- lm(masse_g ~ art + flosse_mm, data =
  pinguine)

# Zuwachs durch die Flossenlänge ...
r2(m1) - 0 # ... wenn sie als Erstes
aufgenommen wird
```

```
[1] 0.7620922
```

```
r2(m_art_flosse) - r2(m_art) # ... wenn die Art bereits
im Modell ist
```

```
[1] 0.1125446
```

Als erster Prädiktor klärt die Flossenlänge 76 % der Varianz auf, nach der Art nur noch 11.3 %. Der Großteil ihrer Vorhersagekraft ist also Varianz, die sie mit der Art teilt.

4.5.6 - Annahmen prüfen

Für die Prüfung der Annahmen bringt R eine eigene Abbildungsfunktion mit. Wendet man `plot()` auf ein `lm`-Objekt an, entstehen vier diagnostische Grafiken: Residuen gegen vorhergesagte Werte, ein Q-Q-Plot der standardisierten Residuen, ein Scale-Location-Plot zur Prüfung der Homoskedastizität und ein Plot der Residuen gegen die Hebelwerte mit Konturlinien für Cooks Distanz.

```
par(mfrow = c(2, 2))
plot(m3)
```

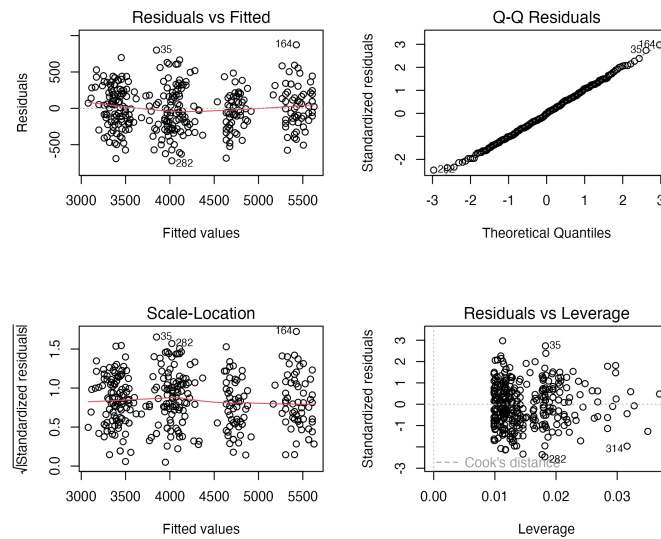
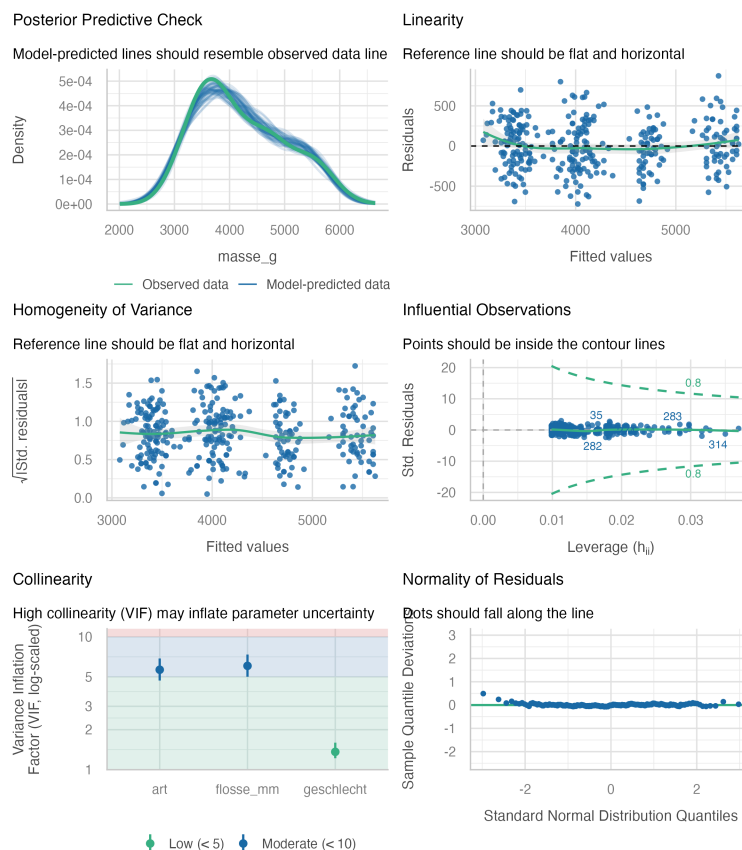


Abbildung 4.7: Die vier diagnostischen Standardgrafiken von `plot()` für das vollständige Modell `m3`.

Übersichtlicher und mit Hinweisen, worauf man achten sollte, ist die Funktion `check_model()` aus dem Paket `performance`. Sie erzeugt ein ganzes Panel von Diagnosegrafiken, die jeweils im Untertitel beschreiben, wie das Idealbild aussieht:

```
library(performance)
check_model(m3)
```

Abbildung 4.8: Diagnosegrafiken mit `performance::check_model()`.**i Hinweis**

`performance` und das für die Grafiken benötigte `see` sind Zusatzpakete: `install.packages(c("performance", "see"))`.

Einzelne Aspekte lassen sich auch gezielt abfragen. Die **Varianzinflationsfaktoren** liefert `check_collinearity()` (oder `vif()` aus dem Paket `car`). Einflussreiche Fälle findet man über **Cooks Distanz**; eine verbreitete Faustregel betrachtet Werte über $4/n$ genauer:

```
check_collinearity(m3)
```

```
# Check for Multicollinearity
```

```
Low Correlation
```

Term	VIF	VIF 95% CI	adj. VIF	Tolerance	Tolerance 95% CI
geschlecht	1.36	[1.22, 1.60]	1.17	0.73	[0.63, 0.82]

```
Moderate Correlation
```

Term	VIF	VIF 95% CI	adj. VIF	Tolerance	Tolerance 95% CI
------	-----	------------	----------	-----------	------------------

```
95% CI
flosse_mm 6.05 [5.01, 7.36]    2.46    0.17    [0.14,
0.20]
art 5.65 [4.69, 6.87]    1.54    0.18    [0.15, 0.21]
```

```
cooks <- cooks.distance(m3)
sum(cooks > 4 / nrow(pinguine))    # Anzahl auffälliger
Fälle
```

```
[1] 13
```

```
pinguine[which.max(cooks), ]    # der einflussreichste
Fall
```

```
# A tibble: 1 × 7
  art      insel schnabel_mm schnabeltiefe_mm flosse_mm
masse_g geschlecht
  <fct>   <fct>      <dbl>          <dbl>      <int>
<int> <fct>
1 Adelie Dream      39.8          19.1      184
4650 männlich
```

Die Faustregel ist bewusst streng und markiert in fast jedem Datensatz einige Fälle. Das ist kein Grund, diese auszuschließen, sondern ein Anlass, sie sich anzusehen und ggf. das Modell zur Kontrolle ohne sie zu schätzen.

Stellt sich Heteroskedastizität heraus, kann man die Standardfehler robust schätzen. Die Pakete `sandwich` und `lmtest` erledigen das zusammen; die Koeffizienten bleiben gleich, nur Standardfehler, t- und p-Werte ändern sich:

```
library(sandwich)
library(lmtest)

coeftest(m3, vcov = vcovHC(m3, type = "HC3"))
```

```
t test of coefficients:
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -365.8174   514.3911 -0.7112  0.47749
flosse_mm      20.0249    2.7514  7.2782 2.518e-12 ***
geschlechtmännlich 530.3811    37.9101 13.9905 < 2.2e-16 ***
***
artChinstrap   -87.6345    50.3464 -1.7406  0.08269 .
artGentoo      836.2600    84.9926  9.8392 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1
```

4.5.7 - Ergebnisse visualisieren

Ganz im Sinne des Kapitels über Abbildungen sollte man sich ein Modell nicht nur als Tabelle ansehen. Für die multiple Regression bietet es sich an, die **vorhergesagten Werte** zusammen mit den Rohdaten darzustellen. Mit `predict()` erhalten wir für jeden Fall den vorhergesagten

Wert, und mit `ggplot2` zeichnen wir ihn als Linie über die Punkte. Weil das Modell keine Interaktionen enthält, entstehen für jede Kombination aus Art und Geschlecht parallele Geraden:

```
penguins |>
  mutate(vorhersage = predict(m3)) |>
  ggplot(aes(flosse_mm, masse_g, colour = art)) +
  geom_point(alpha = 0.4) +
  geom_line(aes(y = vorhersage, linetype = geschlecht),
    linewidth = 0.8) +
  scale_colour_viridis_d(end = 0.85) +
  labs(x = "Flossenlänge (mm)", y = "Körpergewicht (g)",
    colour = "Art", linetype = "Geschlecht")
```

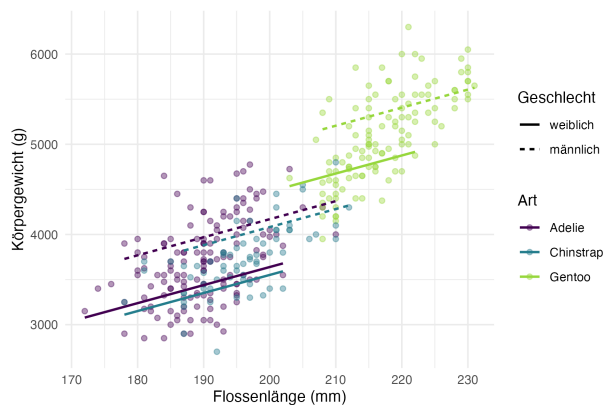


Abbildung 4.9: Vorhergesagte Werte aus Modell 3 (Linien) und Rohdaten (Punkte). Die Geraden sind parallel, weil das Modell für alle Gruppen dieselbe Steigung der Flossenlänge annimmt.

Für einzelne Konstellationen, etwa das erwartete Gewicht eines männlichen Gentoo-Pinguins mit 220 mm Flossenlänge, übergibt man `predict()` einen neuen Datensatz. Mit `interval = "confidence"` erhält man das Konfidenzintervall für den *mittleren* vorhergesagten Wert, mit `interval = "prediction"` das deutlich breitere Vorhersageintervall für ein *einzelnes* Tier:

```
neu <- tibble(flosse_mm = 220, geschlecht = "männlich",
  art = "Gentoo")
```

```
predict(m3, newdata = neu, interval = "confidence")
```

```
      fit      lwr      upr
1 5406.305 5344.53 5468.08
```

```
predict(m3, newdata = neu, interval = "prediction")
```

```
      fit      lwr      upr
1 5406.305 4821.591 5991.019
```

4.6 - Schicke Tabellen mit `modelsummary`

Die Ausgabe von `summary()` ist für die eigene Arbeit gut geeignet, für einen Bericht oder eine Publikation aber unbrauchbar. Regressions-ergebnisse von Hand in eine Word-Tabelle zu übertragen, ist nicht nur mühsam, sondern genau die Art von Handarbeit, vor der uns das Kapitel zum Datenmanagement gewarnt hat: Jeder Tippfehler bleibt unbemerkt, und jede Änderung am Modell erfordert eine neue Abschrift. Das Paket `modelsummary` (Arel-Bundock, 2022) erzeugt aus einem oder mehreren Modellen direkt eine formatierte Tabelle, die sich nach HTML, Word, PDF oder LaTeX exportieren lässt.

i Hinweis

`modelsummary` ist ein Zusatzpaket:
`install.packages("modelsummary")`.

4.6.1 - Die Grundfunktion

Im einfachsten Fall übergeben Sie `modelsummary()` ein Modell oder eine (benannte) Liste von Modellen. Jedes Modell wird zu einer Spalte, gleichnamige Koeffizienten werden in dieselbe Zeile geschrieben, und am Ende stehen die Kennwerte der Modellgüte:

```
library(modelsummary)

modelsummary(modelle)
```

Tabelle 4.5: Hierarchische Regression des Körpergewichts, Standardausgabe von `modelsummary()`.

	Modell 1	Modell 2	Modell 3
(Intercept)	-5872.093 (310.285)	-5410.300 (285.798)	-365.817 (532.050)
flosse_mm	50.153 (1.540)	46.982 (1.441)	20.025 (2.846)
geschlechtmännlich		347.850 (40.342)	530.381 (37.810)
artChinstrap			-87.634 (46.347)
artGentoo			836.260 (85.185)
Num.Obs.	333	333	333
R2	0.762	0.806	0.867
R2 Adj.	0.761	0.805	0.865
AIC	4928.1	4862.5	4740.8
BIC	4939.6	4877.7	4763.6
Log.Lik.	-2461.073	-2427.242	-2364.387
F	1060.295	684.803	534.024
RMSE	392.16	354.28	293.34

Die Namen der Liste werden zu Spaltenüberschriften. Eine unbenannte Liste erhält die Überschriften (1), (2), (3).

Schon diese Standardtabelle ist ein großer Fortschritt: Die drei Modelle der hierarchischen Regression stehen nebeneinander, man sieht auf einen Blick, wie sich die Koeffizienten der Flossenlänge beim Hinzufügen weiterer Prädiktoren verändern, und die Standardfehler stehen in Klammern darunter. Für eine Publikation fehlen aber noch einige Anpassungen.

4.6.2 - Tabellen anpassen

Die wichtigsten Stellschrauben sind die folgenden Argumente:

- `estimate` und `statistic` legen fest, was in den Zellen steht. Beide akzeptieren Vorlagen mit geschweiften Klammern, etwa `"{estimate}{stars}"` für den Koeffizienten mit Signifikanzsternchen oder `"[{conf.low}, {conf.high}]"` für das Konfidenzintervall.
- `stars` bestimmt die Grenzen für die Sternchen. Die Voreinstellung verwendet auch ein + für $p < .10$; mit `c("*" = .05, "***" = .01, "****" = .001)` erhält man die in der Psychologie üblichen Stufen.
- `coef_map` benennt die Koeffizienten um und legt gleichzeitig ihre Reihenfolge fest. Koeffizienten, die nicht aufgeführt sind, werden weggelassen. (Wer nur umbenennen möchte, nimmt `coef_rename`.)

- `gof_map` wählt die Kennwerte der Modellgüte (*goodness of fit*) aus, benennt sie und legt die Nachkommastellen fest.
- `fmt` bestimmt die Zahl der Nachkommastellen,
- `notes` fügt Anmerkungen unter der Tabelle hinzu.

Mit diesen Bausteinen entsteht eine Tabelle, die man so in eine Haus- oder Abschlussarbeit übernehmen kann:

```
koeffizienten <- c(
  "(Intercept)"      = "Achsenabschnitt",
  "flosse_mm"        = "Flossenlänge (mm)",
  "geschlechtmännlich" = "Geschlecht: männlich",
  "artChinstrap"     = "Art: Zügelpinguin",
  "artGentoo"        = "Art: Eselspinguin"
)

guete <- tribble(
  ~raw,      ~clean,      ~fmt,
  "nobs",    "N",          0,
  "r.squared", "R²",        3,
  "adj.r.squared", "R² (korr.)", 3,
  "aic",     "AIC",        0
)

delta_zeile <- tibble(
  term      = "ΔR²",
  `Modell 1` = r2_tabelle$delta_r2[1],
  `Modell 2` = r2_tabelle$delta_r2[2],
  `Modell 3` = r2_tabelle$delta_r2[3]
) |>
  mutate(across(-term, \(x) sprintf("%.3f", x)))
```

```
modelsummary(
  modelle,
  estimate      = "{estimate}{stars}",
  statistic     = "[{conf.low}, {conf.high}]",
  stars         = c("*" = .05, "***" = .01, "****" = .001),
  fmt          = 1,
  notes        = "Referenzkategorien: weiblich,
  Adeliepinguin."
)
```

Tabelle 4.6: Hierarchische Regression des Körpergewichts (g) auf Flossenlänge, Geschlecht und Art. Angegeben sind unstandardisierte Koeffizienten mit 95%-Konfidenzintervallen.

	Modell 1	Modell 2	Modell 3
(Intercept)	-5872.1*** [-6482.5, -5261.7]	-5410.3*** [-5972.5, -4848.1]	-365.8 [-1412.5, 680.8]
flosse_mm	50.2*** [47.1, 53.2]	47.0*** [44.1, 49.8]	20.0*** [14.4, 25.6]
geschlechtmänn- lich		347.9*** [268.5, 427.2]	530.4*** [456.0, 604.8]
artChinstrap			-87.6 [-178.8, 3.5]
artGentoo			836.3*** [668.7, 1003.8]
Num.Obs.	333	333	333
R2	0.762	0.806	0.867
R2 Adj.	0.761	0.805	0.865
AIC	4928.1	4862.5	4740.8
BIC	4939.6	4877.7	4763.6
Log.Lik.	-2461.073	-2427.242	-2364.387
F	1060.295	684.803	534.024
RMSE	392.16	354.28	293.34

Referenzkategorien: weiblich, Adeliepinguin.

Tabelle 4.6 zeigt die Ergebnisse der hierarchischen Regression kompakt und vollständig: Koeffizienten mit Sternchen, Konfidenzintervalle statt Standardfehler, verständliche Beschriftungen, die Varianzaufklärung jedes Modells und den jeweiligen Zuwachs.

4.6.3 - Robuste Standardfehler in der Tabelle

`modelsummary()` kann über das Argument `vcov` auch direkt robuste Standardfehler berechnen. Übergibt man dasselbe Modell zweimal mit unterschiedlichen Schätzern für die Standardfehler, sieht man sofort, ob sich die Schlussfolgerungen ändern oder nicht.

```
modelsummary(
  list("Klassisch" = m3, "Robust (HC3)" = m3),
  vcov      = c("classical", "HC3"),
```

```
fmt      = 1
)
```

Tabelle 4.7: Modell 3 mit klassischen und heteroskedastizitätskonsistenten (HC3) Standardfehlern.

	Klassisch	Robust (HC3)
(Intercept)	-365.8 (532.1)	-365.8 (514.4)
flosse_mm	20.0 (2.8)	20.0 (2.8)
geschlechtmännlich	530.4 (37.8)	530.4 (37.9)
artChinstrap	-87.6 (46.3)	-87.6 (50.3)
artGentoo	836.3 (85.2)	836.3 (85.0)
Num.Obs.	333	333
R2	0.867	0.867
R2 Adj.	0.865	0.865
AIC	4740.8	4740.8
BIC	4763.6	4763.6
Log.Lik.	-2364.387	-2364.387
F	534.024	566.845
RMSE	293.34	293.34
Std.Errors	Classical	HC3

4.6.4 - Koeffizienten als Abbildung

Tabellen sind gut zum Nachschlagen, aber Abbildungen sind besser zum Vergleichen. Die Funktion `modelplot()` aus demselben Paket stellt die Koeffizienten mit ihren Konfidenzintervallen als **Koeffizientenplot** dar. Weil das Ergebnis ein gewöhnliches ggplot-Objekt ist, lässt es sich mit allen Mitteln aus dem Kapitel über Abbildungen weiter anpassen:

```
modelplot(modelle, coef_map = rev(koeffizienten[-1])) +
  geom_vline(xintercept = 0, linetype = "dashed", colour = "grey50") +
  scale_colour_viridis_d(end = 0.85) +
  labs(x = "Koeffizient (g) mit 95%-KI", colour = NULL)
```

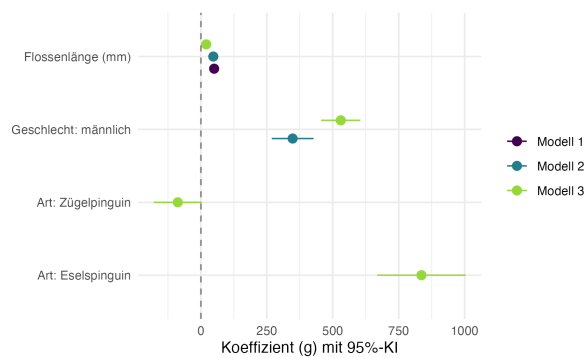


Abbildung 4.10: Koeffizienten der drei Modelle mit 95%-Konfidenzintervallen (modelplot).

Den Achsenabschnitt lässt man im Koeffizientenplot meist weg (dafür `koeffizienten[-1]`), weil er auf einer ganz anderen Skala liegt und alle anderen Koeffizienten zusammenstaucht.

4.6.5 - Deskriptive Tabellen

Neben Modelltabellen bietet `modelsummary` eine Familie von Funktionen für deskriptive Tabellen, die alle mit `datasummary` beginnen. Sie sind eine Alternative zu `compareGroups` aus dem Kapitel zum Datenmanagement und haben den Vorteil, dass sie dieselben Exportwege nutzen. `datasummaryskim()` gibt einen schnellen Überblick über alle numerischen Variablen, `datasummarycorrelation()` erstellt eine Korrelationsmatrix, wie sie in fast jeder Publikation vor den Regressionsergebnissen steht, und `datasummarybalance()` vergleicht Gruppen:

```
penguins |>
  select(Masse = masse_g, Flosse = flosse_mm,
         Schnabellänge = schnabel_mm, Schnabeltiefe =
         schnabeltiefe_mm) |>
  datasummarycorrelation()
```

Tabelle 4.8: Korrelationsmatrix der kontinuierlichen Variablen.

	Masse	Flosse	Schnabellänge	Schnabeltiefe
Masse	1	.	.	.
Flosse	.87	1	.	.
Schnabellänge	.59	.65	1	.
Schnabeltiefe	-.47	-.58	-.23	1

```
penguins_auswahl <- penguins |>
  select(geschlecht, masse_g, flosse_mm, schnabel_mm, art)

datasummarybalance(~ geschlecht, data =
  penguins_auswahl)
```

Tabelle 4.9: Deskriptive Statistiken nach Geschlecht.

		weiblich (N=165)		männlich (N=168)	
		Mean	Std. Dev.	Mean	Std. Dev.
masse_g		3862.3	666.2	4545.7	787.6
flosse_mm		197.4	12.5	204.5	14.5
schnabel_mm		42.1	4.9	45.9	5.4
		N	Pct.	N	Pct.
art	Adelie	73	44.2	73	43.5
	Chinstrap	34	20.6	34	20.2
	Gentoo	58	35.2	61	36.3

4.6.6 - Tabellen exportieren

Alle Funktionen des Pakets haben ein Argument `output`. Gibt man dort einen Dateinamen an, bestimmt die Endung das Format. So landet die Tabelle direkt in einem Word-Dokument, das Sie in Ihre Arbeit einfügen können, ohne eine einzige Zahl abzutippen:

```
modelsummary(modelle, output = "regression.docx")
modelsummary(modelle, output = "regression.html")
modelsummary(modelle, output = "regression.tex")
```

4.7 - Fazit

- Das **Allgemeine Lineare Modell** beschreibt Daten als gewichtete Summe von Prädiktoren plus normalverteilten Fehler. t-Test, Korrelation, ANOVA und ANCOVA sind Spezialfälle davon, die sich nur in der Designmatrix unterscheiden.
- In der **multiplen Regression** sind die Gewichte **partiell**: Sie beschreiben den Zusammenhang unter statistischer Konstanzhaltung der übrigen Prädiktoren. Das ist nicht dasselbe wie experimentelle Kontrolle, und welche Variablen ins Modell gehören, ist eine theoretische Frage.
- Im linearen Modell liefern die **kleinsten Quadrate** und die **Maximum-Likelihood-Schätzung** dieselben Gewichte. Die ML-Schätzung ist aber das allgemeinere Prinzip, auf dem fast alle Verfahren der folgenden Kapitel beruhen.
- Die **inkrementelle Varianzaufklärung** ΔR^2 beantwortet die Frage, was ein Prädiktor über andere hinaus beiträgt. Sie hängt von der Reihenfolge ab, sagt wenig über praktische Relevanz aus und wird durch Messfehler in den Kontrollvariablen systematisch überschätzt.
- Die wichtigsten **Annahmen** sind Validität, Linearität und Unabhängigkeit, nicht die Normalverteilung. Geprüft werden sie vor allem mit Abbildungen der Residuen.
- In R schätzt man lineare Modelle mit `lm()` und der Formelschreibweise, vergleicht geschachtelte Modelle mit `anova()`, prüft Annahmen mit `plot()` oder `performance::check_model()` und erstellt mit `modelsummary()` publikationsreife Tabellen, ohne eine einzige Zahl abzutippen.

4.7.1 - Vertiefung

Das mit Abstand beste moderne Lehrbuch zur Regression ist *Regression and Other Stories* von Gelman, Hill und Vehtari (2020). Es ist konsequent anwendungsorientiert, legt großen Wert auf Abbildungen und Simulationen und ist als PDF frei verfügbar. Der Klassiker für die Verhaltenswissenschaften ist das Buch von Cohen, Cohen, West und Aiken (2003), das insbesondere die hierarchische Regression, kategoriale Prädiktoren und Interaktionen sehr gründlich behandelt. Wer die mathematischen Grundlagen einschließlich der Matrixschreibweise vertiefen möchte, findet bei Fox (2016) eine gut lesbare Darstellung. Die Einsicht, dass die gängigen Tests lineare Modelle sind, hat Jonas Kristoffer Lindeløv (2019) in einer frei verfügbaren Übersicht mit R-Code für jeden einzelnen Test aufbereitet. Und wer mit `modelsummary` noch mehr machen möchte, findet auf der Website des Pakets (<https://modelsummary.com>) eine ausgezeichnete Dokumentation mit vielen Beispielen (Arel-Bundock, 2022).

4.7.2 - Referenzen

5. Varianzanalysen

„To find out what happens to a system when you interfere with it you have to interfere with it (not just passively observe it).“ – George Box (1966)

In diesem Kapitel lernen Sie die Familie der **Varianzanalysen** kennen, das klassische Werkzeug zur Auswertung von Experimenten. Im Kapitel zur multiplen Regression haben wir gesehen, dass die einfaktorielle Varianzanalyse nichts anderes ist als eine Regression auf Dummy-Variablen. Hier bauen wir auf dieser Einsicht auf. Wir beginnen mit der Idee der **Varianzzerlegung**, die dem Verfahren seinen Namen gibt, und damit, wie man nach einem signifikanten F-Test herausfindet, *welche* Gruppen sich unterscheiden. Dann besprechen wir Fälle, bei denen mehrere Einflussfaktoren gleichzeitig betrachtet werden und mit ihnen die **Interaktionen**, mit deren Hilfe man auch Moderationseffekte untersuchen kann. Die **Kovarianzanalyse** (ANCOVA) ergänzt das Modell um eine Kovariate, die **Varianzanalyse mit Messwiederholung** berücksichtigt, dass dieselben Personen/Pinguine mehrfach gemessen werden. Beides führt uns zu einer Frage, die in der angewandten Forschung ständig auftaucht: Wie wertet man eine Studie aus, in der zwei Gruppen vor und nach einer Intervention gemessen werden, und wie berechnet man dort eine Effektstärke? Zum Schluss des konzeptuellen Teils geht es um die **Fallzahlplanung**. Wie in den vorigen Kapiteln zeigt die zweite Hälfte, wie Sie all das in R umsetzen.

5.1 - Die Idee der Varianzzerlegung

Die Varianzanalyse (*analysis of variance*, ANOVA) wurde in den 1920er-Jahren von Ronald Fisher (1925) für im wahrsten Sinne des Wortes Feldversuche mit Gemüse auf dem Acker entwickelt. Er musste herausfinden, ob sich die Erträge von Parzellen, die mit verschiedenen Düngern behandelt wurden, unterscheiden? Der Name Varianzanalyse ist auf den ersten Blick verwirrend, denn sie testet Unterschiede zwischen **Mittelwerten**. Sie tut das aber, indem sie die **Varianz** der Daten in Anteile zerlegt.

Nehmen wir als Beispiel wieder das Körpergewicht der drei Pinguinarten. Die Werte der einzelnen Tiere weichen mehr oder weniger stark vom Gesamtmittelwert aller Tiere ab (Abbildung 5.1, links). Wenn man die einzelnen Abweichungen der einzelnen Pinguine quadriert und aufsummiert, erhält man einen Wert für die Varianz des Körpergewichts (aka eine Quadratsumme QS). Diese Abweichung lässt sich in zwei Teile aufteilen: die Abweichung des Gruppenmittelwerts vom Gesamtmittelwert, die auf die Art zurückgeht, und die Abweichung des einzelnen Tiers von seinem Gruppenmittelwert, die die Art nicht erklären kann (Abbildung 5.1, rechts). Quadriert und über alle Fälle summiert, gilt für diese Quadratsummen:

$$\underbrace{\sum_j \sum_i (y_{ij} - \bar{y})^2}_{\text{QS}_{\text{gesamt}}} = \underbrace{\sum_j n_j (\bar{y}_j - \bar{y})^2}_{\text{QS}_{\text{zwischen}}} + \underbrace{\sum_j \sum_i (y_{ij} - \bar{y}_j)^2}_{\text{QS}_{\text{innerhalb}}} \quad (5.1)$$

Sie kennen diese Zerlegung aus dem Regressionskapitel: QS_{zwischen} ist die durch das Modell erklärte Quadratsumme, $QS_{\text{innerhalb}}$ die Residuenquadratsumme.

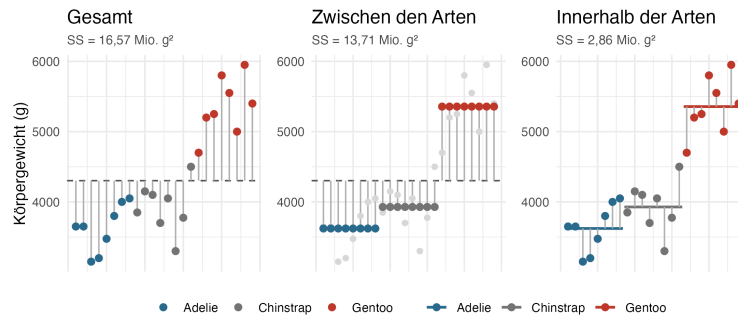


Abbildung 5.1: Varianzzerlegung am Beispiel von je acht zufällig ausgewählten Pinguinen pro Art. Links: Abweichungen der einzelnen Tiere vom Gesamtmittelwert (gestrichelt), deren Quadratsumme QS_{gesamt} ist. Mitte: Abweichungen der Artmittelwerte vom Gesamtmittelwert, eingezeichnet für jedes Tier seiner Art (QS_{zwischen}); die grauen Punkte zeigen die ursprünglichen Werte. Rechts: Abweichungen der Tiere von den Mittelwerten ihrer Art ($QS_{\text{innerhalb}}$). Die Quadratsummen in der Mitte und rechts ergeben zusammen die Gesamtquadratsumme links.

Teilt man die Quadratsummen durch ihre Freiheitsgrade, erhält man mittlere Quadratsummen (*mean squares*, MS). Ihr Verhältnis ist die Prüfgröße der Varianzanalyse, der F-Wert:

$$F = \frac{MS_{\text{zwischen}}}{MS_{\text{innerhalb}}} = \frac{QS_{\text{zwischen}}/(k-1)}{QS_{\text{innerhalb}}/(N-k)} \quad (5.2)$$

Dabei ist k die Zahl der Gruppen und N die Gesamtstichprobe. Gilt die Nullhypothese, dass alle Gruppenmittelwerte gleich sind, sollten beide Quadratsummen gleich groß sein, und F liegt um 1. Je größer die Unterschiede zwischen den Gruppen im Verhältnis zur Streuung innerhalb der Gruppen, desto größer wird F .

Effektstärken. Der Anteil der erklärten Quadratsumme an der Gesamtquadratsumme heißt η^2 (Eta-Quadrat) und ist in der einfaktoriellen ANOVA nichts anderes als das R^2 der entsprechenden Regression. Wie das R^2 überschätzt η^2 den Anteil in der Population leicht; das weniger verzerrte ω^2 (Omega-Quadrat) korrigiert dafür, ähnlich wie das korrigierte R^2 . In mehrfaktoriellen Designs berichtet man meist das **partielle** $\eta_p^2 = QS_{\text{Effekt}} / (QS_{\text{Effekt}} + QS_{\text{Fehler}})$, das die Varianz der übrigen Faktoren herausrechnet. Weil es damit vom Design abhängt, lässt es sich schlecht zwischen Studien vergleichen; das **generalisierte** η_G^2 wurde entwickelt, um dieses Problem zu lösen (Lakens, 2013).

5.1.1 - Nach dem F-Test: Kontraste und Post-hoc-Tests

Ein signifikanter F-Test sagt nur, dass sich *irgendwelche* Gruppen unterscheiden, nicht aber welche. Dafür gibt es zwei Wege.

Geplante Kontraste. Hat man vor der Datenerhebung spezifische Hypothesen, kann man diese gezielt als **Kontraste** prüfen, indem man gewichte beschreibt, die diesen Kontrasten entsprechen. Die Hypothese

„Gentoo-Pinguine sind schwerer als die beiden anderen Arten“, entspricht etwa den Gewichten $(-\frac{1}{2}, -\frac{1}{2}, 1)$ für Adelie, Chinstrap und Gentoo. Geplante Kontraste sind der eleganteste Weg, genau die Frage zu beantworten, die man hat. Streng genommen braucht man dann nicht einmal den F-Test.

Post-hoc-Tests. Hat man keine spezifischen Hypothesen, muss man alle Paare von Gruppen miteinander vergleichen. Bei k Gruppen sind das $k(k-1)/2$ Vergleiche, bei fünf Gruppen also schon zehn. Führt man jeden mit $\alpha = .05$ durch, steigt die Wahrscheinlichkeit, mindestens einen Fehler erster Art zu begehen, deutlich über 5 %. Post-hoc-Tests korrigieren diese Inflation des Alpha-Fehlers dafür: Der **Tukey-Test** ist speziell für alle paarweisen Vergleiche gemacht, die **Holm-Korrektur** ist ein allgemeines und stets besseres Verfahren als die bekannte, aber zu strenge Bonferroni-Korrektur.

5.2 - Mehrfaktorielle Varianzanalyse und Interaktionen

In einem Experiment interessiert oft nicht nur ein Faktor. Angenommen, ein Unternehmen testet ein neues Kommunikationstraining und möchte wissen, ob es bei erfahrenen und unerfahrenen Führungskräften gleich gut wirkt. Das Design hat dann zwei Faktoren mit je zwei Stufen (Bedingung: Kontrolle vs. Training; Erfahrung: gering vs. hoch), also vier Zellen. Ein solches 2X2 Design wertet man typischerweise mit einer **zweifaktoriellen Varianzanalyse** aus. Diese kann drei Effekte prüfen, die jeweils für unterschiedliche Fragen stehen:

- Den **Haupteffekt** der Bedingung: Unterscheiden sich Training und Kontrolle, gemittelt über beide Erfahrungsstufen?
- Den **Haupteffekt** der Erfahrung: Unterscheiden sich erfahrene und unerfahrene Führungskräfte, gemittelt über beide Bedingungen?
- Die **Interaktion**: Hängt der Effekt der Bedingung von der Erfahrung ab?

Die Interaktion ist oft die interessanteste der drei Fragen. Abbildung 5.2 zeigt die drei typischen Muster. Verlaufen die Linien parallel, gibt es keine Interaktion: Das Training wirkt bei beiden Gruppen gleich stark. Bei einer **ordinalen** Interaktion wirkt es in beiden Gruppen in dieselbe Richtung, aber unterschiedlich stark. Bei einer **disordinalen** (hybriden oder gekreuzten) Interaktion kehrt sich der Effekt der Intervention zwischen den erfahrenen und weniger erfahrenen Mitarbeitern sogar um.

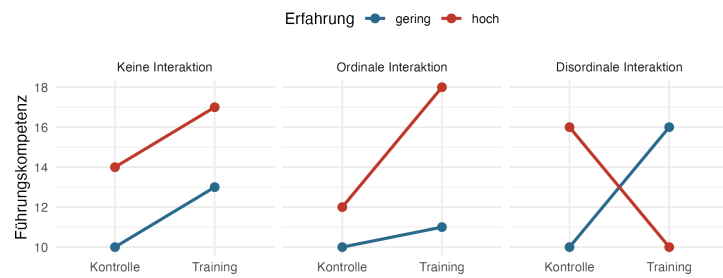


Abbildung 5.2: Drei typische Muster in einem 2×2 -Design. Links: zwei Haupteffekte, keine Interaktion (parallele Linien). Mitte: ordinale Interaktion; das Training wirkt bei erfahrenen Führungskräften stärker. Rechts: disordinale Interaktion; das Training hilft unerfahrenen und schadet erfahrenen Führungskräften.

Interaktionen als Produktterme. Im Regressionsmodell ist eine Interaktion schlicht das Produkt zweier Prädiktoren. Mit zwei Dummy-Variablen A und B lautet das Modell

$$y_i = \beta_0 + \beta_1 A_i + \beta_2 B_i + \beta_3 (A_i \cdot B_i) + \varepsilon_i. \quad (5.3)$$

Der Koeffizient β_3 gibt an, um wie viel sich der Effekt von A verändert, wenn B von 0 auf 1 wechselt, er ist eine **Differenz von Differenzen**. Daraus folgt eine Warnung, die viele Anwenderinnen überrascht: Sobald eine Interaktion im Modell ist, ist β_1 *kein* Haupteffekt mehr, sondern der Effekt von A an der Stelle $B = 0$, also nur in der Referenzgruppe. Man nennt das einen **einfachen Effekt** (*simple effect*).

⚠ Kodierung und Typen von Quadratsummen

Ob die Koeffizienten und Tests für die „Haupteffekte“, das bedeuten, was man erwartet, hängt von zwei technischen Entscheidungen ab.

Erstens von der **Kodierung** der Faktoren. Bei der in R voreingestellten Dummy-Kodierung (`contr.treatment`) beziehen sich die Koeffizienten der Haupteffekte auf die Referenzgruppe des jeweils anderen Faktors. Bei der **Effektkodierung** (`contr.sum`, Werte -1 und $+1$) beziehen sie sich auf den Mittelwert über alle Stufen des anderen Faktors, und das ist es, was typischerweise eher unter einem Haupteffekt versteht.

Zweitens, in Designs mit ungleichen Zellbesetzungen, gibt es unterschiedliche Möglichkeiten (aka Typen) die einzelnen Quadratsummen innerhalb und zwischen für die verschiedenen Faktoren zu berechnen. R berechnet standardmäßig Typ-I-Quadratsummen, die sequenziell sind. Das bedeutet, dass die Quadratsummen eines Faktors davon abhängen, an welcher Stelle er in der Formel steht. Es gibt aber auch sog. Typ-III -Quadratsummen, die jeden Effekt für alle anderen adjustieren und mit Effektkodierung die klassischen Haupteffekte testen. Welcher Typ vorzuziehen ist, wird seit Jahrzehnten diskutiert (Fox, 2016; Maxwell et al., 2018). Für die Praxis genügt: Bei balancierten Designs spielt es keine Rolle; bei unbalancierten sollten Sie wissen, welchen Typ Ihre Software verwendet, und ihn berichten.

Interaktionen sind deutlich schwerer als Haupteffekte zu schätzen. Weil eine Interaktion eine Differenz von Differenzen ist, beruht

jede der beiden Differenzen nur auf der Hälfte der Stichprobe. In einem 2×2 -Design mit gleich großen Zellen ist der Standardfehler der Interaktion deshalb doppelt so groß wie der eines Haupteffekts (vgl. Gelman et al., 2020). Dadurch benötigt man 4-mal so viele Teilnehmer um einen gleich großen Interaktions wie Haupteffekt zu schätzen. Ist die Interaktion zusätzlich nur halb so groß wie der Haupteffekt, was bei ordinalen Interaktionen realistisch ist, benötigt man die **16-fache Stichprobe**, um sie mit derselben Teststärke nachzuweisen. Viele publizierte Interaktionseffekte beruhen auf Studien, die dafür viel zu klein waren.

5.2.1 - Interaktionen mit kontinuierlichen Prädiktoren: Moderation

Nichts an Gleichung (5.3) setzt voraus, dass A und B Dummy-Variablen sind. Ist einer der beiden Prädiktoren kontinuierlich, etwa die Arbeitsbelastung x , und der andere eine Gruppenvariable wie die Teilnahme an einem Resilienztraining z , dann beschreibt β_3 , um wie viel sich die **Steigung** von x zwischen den Gruppen unterscheidet. Man sagt, z **moderiert** den Zusammenhang zwischen x und y (Aiken & West, 1991): Hängt das Stresserleben bei trainierten Beschäftigten weniger stark von der Arbeitsbelastung ab als bei untrainierten, ist das Training ein Moderator. Sind beide Prädiktoren kontinuierlich, gilt dasselbe, nur dass sich die Steigung von x dann kontinuierlich mit z verändert.

Für die Interpretation gelten zwei Regeln. Erstens sollte man kontinuierliche Prädiktoren vor der Bildung des Produktterms **zentrieren** (den Mittelwert abziehen). Der Koeffizient von x ist dann die Steigung bei einem durchschnittlichen Wert von z , statt bei $z = 0$, was oft gar nicht vorkommt. Am Test der Interaktion ändert die Zentrierung nichts, aber es macht die Haupteffekte leichter zu interpretieren. Zweitens interpretiert man eine signifikante Interaktion über **einfache Steigungen** (*simple slopes*): die Steigung von x in jeder Gruppe beziehungsweise bei ausgewählten Werten von z , etwa eine Standardabweichung unter und über dem Mittelwert. Eine Abbildung der Regressionsgeraden für die verschiedenen Werte des Moderators ist dabei fast immer aufschlussreicher als eine Tabelle.

5.3 - Kovarianzanalyse (ANCOVA)

Nimmt man in eine Varianzanalyse zusätzlich einen kontinuierlichen Prädiktor auf, spricht man von einer **Kovarianzanalyse** (*analysis of covariance*, ANCOVA), und den kontinuierlichen Prädiktor nennt man **Kovariate**. Das Modell ist eine gewöhnliche multiple Regression:

$$y_i = \beta_0 + \beta_1 \cdot \text{Gruppe}_i + \beta_2 x_i + \varepsilon_i \quad (5.4)$$

Der Koeffizient β_1 ist die Differenz der Gruppen bei gleichem Wert der Kovariate. Grafisch ist das der vertikale Abstand zwischen zwei **parallelen** Regressionsgeraden (Abbildung 5.3). Die Gruppenmittelwerte, die das Modell für einen mittleren Wert der Kovariate vorhersagt, heißen **adjustierte Mittelwerte** (*estimated marginal means*).

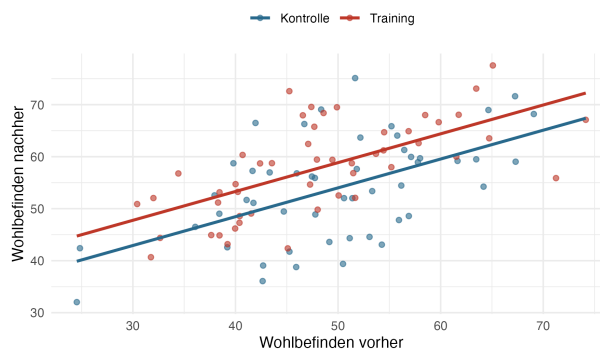


Abbildung 5.3: Kovarianzanalyse in einer simulierten Interventionsstudie: Wohlbefinden nach dem Training in Abhängigkeit vom Wohlbefinden vorher. Der Effekt des Trainings ist der vertikale Abstand zwischen den beiden parallelen Geraden. Die Streuung um die Geraden ist viel kleiner als die Streuung der Nachher-Werte insgesamt.

Die ANCOVA hat, je nach Studiendesign, zwei ganz unterschiedliche Funktionen.

In randomisierten Experimenten erhöht die Kovariate die **Power**, d.h. sie macht es wahrscheinlicher, dass man einen bestehenden Effekt auch per Signifikanztest als solchen erkennt. Da die Gruppen durch die Zufallszuweisung im Mittel gleich sind, verändert die Kovariate den erwarteten Gruppeneffekt nicht. Sie erklärt aber einen Teil der Fehlervarianz: Korreliert die Kovariate innerhalb der Gruppen zu ρ mit dem Kriterium, schrumpft die Fehlervarianz ungefähr um den Faktor $1 - \rho^2$. Bei $\rho = .6$ sind das 36 % weniger Fehlervarianz, was der Teststärke einer um gut die Hälfte größeren Stichprobe entspricht. Die Kovariate muss dafür *vor* der Intervention gemessen worden sein; eine Kovariate, die selbst von der Intervention beeinflusst wird, kann den Effekt verzerren.

In nicht randomisierten Studien soll die Kovariate dagegen Unterschiede zwischen den Gruppen „herausrechnen“. Das ist weitaus heikler. Gruppen, die sich in einer Kovariate unterscheiden, unterscheiden sich meist auch in vielen anderen Merkmalen, und Messfehler in der Kovariate führen dazu, dass die Adjustierung unvollständig bleibt (Miller & Chapman, 2001).

Die ANCOVA setzt voraus, dass die Geraden parallel sind (**Homogenität der Regressionssteigungen**). Das lässt sich prüfen, indem man die Interaktion Gruppe $\times$ Kovariate ins Modell aufnimmt. Ist sie deutlich, ist der Gruppeneffekt nicht mehr eine fixe Zahl, sondern hängt vom Wert der Kovariate ab. Die Kovariate ist eine Moderatorvariable.

5.4 - Varianzanalyse mit Messwiederholung

In vielen psychologischen Studien werden dieselben Personen mehrfach gemessen: vor und nach einer Intervention, unter verschiedenen experimentellen Bedingungen oder zu mehreren Zeitpunkten. Die Messungen einer Person sind dann nicht unabhängig voneinander. Die Ergebnisse einer gewöhnlichen ANOVA wären damit falsch. Die **Varianzanalyse mit Messwiederholung** (*repeated measures ANOVA*, *rmANOVA*) nutzt diese Abhängigkeit zu ihrem Vorteil. Sie zerlegt die Varianz innerhalb der Bedingungen noch einmal in zwei Teile: die stabilen Unterschiede **zwischen Personen** (manche Menschen haben generell ein höheres Wohlbefinden als andere) und die **Restvarianz**. Nur die Restvarianz geht in den Fehlerterm des F-Tests für den Messwiederholungsfaktor ein. Weil stabile Personenunterschiede

in der Psychologie meist einen großen Teil der Varianz ausmachen, haben Messwiederholungsdesigns oft eine erheblich höhere Power als Designs, in denen jede Person nur einer Bedingung ausgesetzt wird.

Sphärizität. Die rmANOVA hat eine zusätzliche Voraussetzung: Die Varianzen aller paarweisen Differenzen zwischen den Messzeitpunkten müssen gleich sein. Bei nur zwei Messzeitpunkten gibt es nur eine Differenz, und die Annahme ist automatisch erfüllt. Bei drei und mehr Zeitpunkten ist sie oft verletzt, weil benachbarte Zeitpunkte stärker korrelieren als weit auseinanderliegende. Der F-Test wird dann zu liberal. Die übliche Lösung ist die **Greenhouse-Geisser-Korrektur** (Greenhouse & Geisser, 1959), die die Freiheitsgrade mit einem Faktor $\varepsilon \leq 1$ multipliziert, der das Ausmaß der Verletzung widerspiegelt. Da die Korrektur bei erfüllter Annahme kaum schadet, empfiehlt es sich, diese Korrektur immer anzuwenden, statt die Entscheidung vom wenig zuverlässigen Mauchly-Test abhängig zu machen.

Gemischte Designs. Kombiniert man einen Messwiederholungsfaktor mit einem Gruppenfaktor, erhält man ein **gemischtes Design** (*mixed design* oder *split-plot design*) bei dem eben within (die Zeit) und between Effekte (die Gruppe) gleichzeitig betrachtet werden. Das wichtigste Beispiel ist die Interventionsstudie mit Kontroll- und Trainingsgruppe, die vor und nach dem Training gemessen werden. Hier interessiert vor allem die Interaktion Gruppe $\times$ Zeit: Verändert sich die Trainingsgruppe stärker als die Kontrollgruppe?

Die klassische rmANOVA schließt Personen mit auch nur einem fehlenden Messzeitpunkt vollständig aus und setzt voraus, dass alle Personen zu denselben Zeitpunkten gemessen wurden.

5.5 - Prä-post-Designs mit zwei Gruppen

Die randomisierte Studie mit Vorher- und Nachhermessung ist das Standarddesign der Interventionsforschung. Umso erstaunlicher ist, wie viele verschiedene Auswertungen man in der Literatur findet. Die vier häufigsten sind:

1. **Nur die Nachhermessung** vergleichen (t-Test oder ANOVA auf Post-Werte).
2. **Veränderungswerte** vergleichen: Für jede Person die Differenz Post minus Prä bilden und die Differenzen zwischen den Gruppen vergleichen.
3. Eine **rmANOVA** mit dem Gruppenfaktor, dem Messwiederholungsfaktor Zeit und ihrer Interaktion. Der Test der Interaktion ist dabei mathematisch identisch mit dem Test der Veränderungswerte: Der F-Wert der Interaktion ist das Quadrat des t-Werts aus Variante 2 und liefert daher auch exakt dieselben p-Werte.
4. Eine **ANCOVA** mit den Post-Werten als Kriterium, der Gruppe als Faktor und den Prä-Werten als Kovariate.

In einer randomisierten Studie schätzen alle vier Varianten denselben Effekt ohne Verzerrung. Sie unterscheiden sich aber erheblich in ihrer **Präzision** und damit in der Power. Nimmt man an, dass Prä- und Post-Werte dieselbe Standardabweichung SD haben und innerhalb der Gruppen zu ρ korrelieren, ist die Varianz des geschätzten Effekts proportional zu

$$\underbrace{1}_{\text{nur Post}}, \quad \underbrace{2(1-\rho)}_{\text{Veränderungswerte oder rmANOVA}}, \quad \underbrace{1-\rho^2}_{\text{ANCOVA}}. \quad (5.5)$$

Daraus folgen drei Einsichten, die in Abbildung 5.4 dargestellt sind. Erstens sind Veränderungswerte nur dann präziser als die reine Nachhermessung, wenn $\rho > .5$ ist; bei kleineren Korrelationen fügt die Differenzbildung mehr Messfehler hinzu, als sie an Personenunterschieden

entfernt. Zweitens ist die ANCOVA **immer mindestens so präzise** wie die anderen Alternativen, denn $1 - \rho^2 \leq 1$ und $1 - \rho^2 \leq 2(1 - \rho)$ für alle ρ . Drittens ist der Vorteil der ANCOVA gerade bei mittleren Korrelationen erheblich: Bei $\rho = .5$ braucht sie ein Viertel weniger Personen als die beiden anderen Varianten.

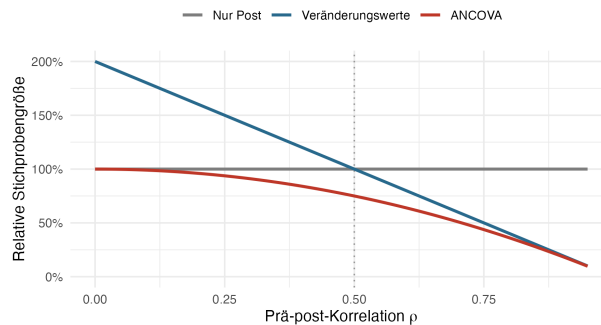


Abbildung 5.4: Benötigte Stichprobengröße für dieselbe Teststärke, relativ zu einer Studie, die nur die Nachhermessung auswertet, in Abhängigkeit von der Prä-post-Korrelation ρ . Die ANCOVA ist bei jeder Korrelation mindestens so effizient wie die beiden Alternativen.

Die ANCOVA hat außerdem einen zweiten Vorteil: Sie korrigiert für **zufällige Unterschiede in der Ausgangslage**. Auch in randomisierten Studien sind die Mitglieder der Trainingsgruppe vor der Intervention zufällig mal etwas besser, mal etwas schlechter als die Kontrollgruppe. Der reine Post-Vergleich ignoriert das, und Veränderungswerte überkorrigieren es wegen der Regression zur Mitte: Eine Gruppe, die zufällig schlecht startet, verbessert sich auch ohne Intervention etwas. Die ANCOVA adjustiert genau im richtigen Ausmaß. Aus diesen Gründen empfehlen Methodiker die ANCOVA seit Langem als Standardauswertung randomisierter Prä-post-Studien (Senn, 2006; Van Breukelen, 2006; Vickers & Altman, 2001). Die Einschränkung aus dem vorigen Abschnitt gilt aber auch hier: In nicht randomisierten Studien kann die ANCOVA verzerrt sein, und die Wahl der Analyse ist dort eine inhaltliche Entscheidung.

⚠ Ein häufiger Fehler: getrennte Tests in beiden Gruppen

In vielen Arbeiten findet man folgende Argumentation: „In der Trainingsgruppe stieg das Wohlbefinden signifikant ($p = .01$), in der Kontrollgruppe nicht ($p = .20$). Das Training ist also wirksam. „ Diese Schlussfolgerung ist nicht gerechtfertigt. Getestet wurde zweimal, ob sich eine Gruppe verändert, aber nie, ob sich die Gruppen *unterschiedlich* verändern. Der Unterschied zwischen „signifikant“ und „nicht signifikant“, ist selbst nicht notwendigerweise signifikant (Gelman & Stern, 2006). Gerade bei kleinen Studien und einem allgemeinen Zeiteffekt führt dieses Vorgehen häufig zu falschen Schlüssen (Bland & Altman, 2011). Getestet werden muss immer der Unterschied zwischen den Gruppen, mit einer der vier oben beschriebenen Methoden, am besten der ANCOVA.

5.5.1 - Effektstärken im Prä-post-Design

Auch bei der Effektstärke gibt es in Prä-post-Designs mehrere Möglichkeiten und einige Fallstricke. Cohens d ist eine Mittelwertsdifferenz geteilt durch eine Standardabweichung. Die entscheidende Frage ist, *welche* Standardabweichung. Damit d zwischen Studien mit verschiedenen Designs vergleichbar ist, sollte es sich auf die Streuung der Werte in der Population beziehen, also auf die Standardabweichung der **Rohwerte** und nicht auf die der Veränderungswerte.

Morris (2008) hat verschiedene Schätzer verglichen und empfiehlt, die Differenz der mittleren Veränderungen durch die gepoolte Standardabweichung der **Prä-Werte** zu teilen:

$$d_{\text{ppc}} = c_p \cdot \frac{(\bar{y}_{T,\text{post}} - \bar{y}_{T,\text{prä}}) - (\bar{y}_{K,\text{post}} - \bar{y}_{K,\text{prä}})}{SD_{\text{prä, gepoolt}}}, \quad c_p = 1 - \frac{3}{4(n_T + n_K - 2) - 1}. \quad (5.6)$$

T steht für die Trainings-, K für die Kontrollgruppe, und der Faktor c_p korrigiert eine kleine Verzerrung in kleinen Stichproben. Die Prä-Werte haben den Vorteil, dass sie von der Intervention noch nicht beeinflusst sind. Wertet man mit einer ANCOVA aus, teilt man entsprechend die adjustierte Mittelwertsdifferenz durch die gepoolte Standardabweichung der Rohwerte.

Teilt man dagegen durch die Standardabweichung der **Veränderungswerte**, erhält man eine andere Größe, die oft mit d_z oder d_{rm} bezeichnet wird. Weil die Veränderungswerte bei hoher Prä-post-Korrelation viel weniger streuen, fällt dieses d um den Faktor $1/\sqrt{2(1-\rho)}$ größer aus, bei $\rho = .8$ also um mehr als das Eineinhalbfache. Es ist nicht falsch, aber nicht mit einem d aus einem Design ohne Messwiederholung vergleichbar. Wer es berichtet, ohne das deutlich zu machen, lässt einen Effekt größer erscheinen, als er ist.

5.6 - Fallzahlplanung

Die **Teststärke** (Power) eines Tests ist die Wahrscheinlichkeit, einen Effekt einer bestimmten Größe zu entdecken, wenn es ihn gibt. Sie hängt von vier Größen ab, von denen man drei festlegen muss, um die vierte zu berechnen: dem Signifikanzniveau α , der Teststärke $1 - \beta$, der Effektgröße und der Stichprobengröße. Wie wir in der Einleitung gesehen haben, sind viele psychologische Studien viel zu klein (Cohen, 1988). Eine Fallzahlplanung vor der Datenerhebung ist deshalb heute Standard und wird von Ethikkommissionen, Geldgebern und Zeitschriften für Präregistrierungen verlangt.

Für den Vergleich zweier Gruppen gibt es eine nützliche Faustformel. Mit $\alpha = .05$ (zweiseitig) und einer Teststärke von 80 % braucht man pro Gruppe ungefähr

$$n \approx \frac{2(z_{1-\alpha/2} + z_{1-\beta})^2}{d^2} \approx \frac{16}{d^2} \quad (5.7)$$

Personen (Lehr, 1992). Für einen mittleren Effekt von $d = 0,5$ sind das 64, für einen kleinen Effekt von $d = 0,2$ schon 400 Personen pro Gruppe. Diese Formel lässt sich direkt auf die Auswertungsvarianten des Prä-post-Designs übertragen: Man multipliziert die Stichprobe mit den Faktoren aus Abbildung 5.4, für die ANCOVA also mit $1 - \rho^2$ (Borm et al., 2007). Tabelle 5.1 zeigt die Konsequenzen für eine realistische Prä-post-Korrelation von $\rho = .6$.

Tabelle 5.1: Benötigte Stichprobengröße pro Gruppe für eine Teststärke von 80 % ($\alpha = .05$, zweiseitig) in einer randomisierten Prä-post-Studie mit $\rho = .6$, je nach Auswertungsmethode (Normalverteilungsnaherung; exakte Werte liegen um ein bis zwei Personen höher als nach der Daumenregel).

Effekt d	Nur Post	Veränderungswerte	ANCOVA
0,20	393	314	252
0,35	129	103	83
0,50	63	51	41
0,80	25	20	16

Die schwierigste Frage bei jeder Fallzahlplanung ist, welche Effektgröße man annehmen soll. Drei Ratschläge helfen weiter (Lakens, 2022):

- **Keine Effektgrößen aus Pilotstudien.** Kleine Pilotstudien schätzen Effekte so ungenau, dass eine darauf beruhende Planung oft viel zu kleine Stichproben ergibt (Albers & Lakens, 2018).
- **Den kleinsten relevanten Effekt planen.** Statt zu fragen, wie groß der Effekt vermutlich ist, fragt man, wie groß er mindestens sein müsste, damit er praktisch bedeutsam ist. Für ein Training, das pro Person 2.000 € kostet, kann das ein ganz anderer Wert sein als für eine kostenlose App.
- **Das Auswertungsmodell zugrunde legen.** Die Fallzahl muss für den Test berechnet werden, der später tatsächlich gerechnet wird. Wer eine ANCOVA plant, aber für einen t-Test rechnet, verschenkt Ressourcen; wer eine Interaktion testen will, aber für einen Haupteffekt plant, hat eine viel zu kleine Studie.

Für Standarddesigns gibt es Formeln und Programme wie G*Power (Faul et al., 2007) oder das R-Paket `pwr`. Für komplexere Designs, etwa rmANOVAs mit mehreren Messzeitpunkten, Interaktionen oder Mehrebenenmodelle, ist die **Simulation** der flexibelste Weg: Man erzeugt viele Datensätze mit dem angenommenen Effekt, wertet jeden so aus wie geplant und zählt, wie oft der Test signifikant wird.

5.7 - Varianzanalysen mit R

Nach so viel Konzept geht es nun an die praktische Umsetzung. Einfache Varianzanalysen können Sie, wie im Regressionskapitel gezeigt, mit `lm()` und `anova()` rechnen. Für mehrfaktorielle und Messwiederholungsdesigns verwenden wir zusätzlich drei spezialisierte Pakete: `afex` für die Schätzung mit den üblichen Voreinstellungen (Effektkodierung, Typ-III-Quadratsummen, Greenhouse-Geisser-Korrektur), `emmeans` für adjustierte Mittelwerte, Kontraste und Post-hoc-Tests und `effectsize` für Effektstärken (Ben-Shachar et al., 2020).

5.7.1 - Beispieldatensatz

```
library(tidyverse)
library(palmerpenguins)
library(afex)
library(emmeans)
```

```

pinguine <- penguins |>
  select(art = species, insel = island,
         schnabel_mm = bill_length_mm, schnabeltiefe_mm
         = bill_depth_mm,
         flosse_mm = flipper_length_mm, masse_g =
         body_mass_g,
         geschlecht = sex) |>
  drop_na() |>
  mutate(geschlecht = fct_recode(geschlecht,
                                weiblich = "female",
                                männlich = "male"),
         id = row_number())

```

afex braucht eine Variable, die jede Untersuchungseinheit eindeutig kennzeichnet, auch wenn es gar keine Messwiederholung gibt. Deshalb legen wir mit `row_number()` eine `id` an.

5.7.2 - Einfaktorielle Varianzanalyse

Die einfaktorielle ANOVA kennen Sie bereits aus dem Regressionskapitel. Wir schätzen sie noch einmal mit `lm()` und berechnen die Effektstärken η^2 und ω^2 :

```

m_1f <- lm(masse_g ~ art, data = pinguine)
anova(m_1f)

```

Analysis of Variance Table

```

Response: masse_g
      Df Sum Sq Mean Sq F value    Pr(>F)
art      2 145190219 72595110  341.89 < 2.2e-16 ***
Residuals 330  70069447  212332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

```

```
effectsize::eta_squared(m_1f)
```

Effect Size for ANOVA

```

Parameter | Eta2 |      95% CI
-----
art      | 0.67 | [0.63, 1.00]

```

- One-sided CIs: upper bound fixed at [1.00].

```
effectsize::omega_squared(m_1f)
```

Effect Size for ANOVA

```

Parameter | Omega2 |      95% CI
-----
art      | 0.67 | [0.63, 1.00]

```

- One-sided CIs: upper bound fixed at [1.00].

Welche Arten sich unterscheiden, zeigen die **paarweisen Vergleiche**. `emmeans()` berechnet zuerst die Mittelwerte jeder Art aus dem Modell; `pairs()` vergleicht sie und korrigiert dabei standardmäßig nach Tukey:

```
emm_art <- emmeans(m_1f, ~ art)
emm_art
```

art	emmean	SE	df	lower.CL	upper.CL
Adelie	3706	38.1	330	3631	3781
Chinstrap	3733	55.9	330	3623	3843
Gentoo	5092	42.2	330	5009	5176

Confidence level used: 0.95

```
pairs(emm_art)
```

contrast	estimate	SE	df	t.ratio	p.value
Adelie - Chinstrap	-26.9	67.7	330	-0.398	0.9164
Adelie - Gentoo	-1386.3	56.9	330	-24.359	<0.0001
Chinstrap - Gentoo	-1359.3	70.0	330	-19.406	<0.0001

P value adjustment: tukey method for comparing a family of 3 estimates

```
pairs(emm_art, adjust = "holm")
```

contrast	estimate	SE	df	t.ratio	p.value
Adelie - Chinstrap	-26.9	67.7	330	-0.398	0.6909
Adelie - Gentoo	-1386.3	56.9	330	-24.359	<0.0001
Chinstrap - Gentoo	-1359.3	70.0	330	-19.406	<0.0001

P value adjustment: holm method for 3 tests

Einen **geplanten Kontrast** übergibt man `contrast()` als benannte Liste von Gewichten in der Reihenfolge der Faktorstufen:

```
contrast(emm_art, list("Gentoo vs. andere" = c(-1/2, -1/2, 1)))
```

contrast	estimate	SE	df	t.ratio	p.value
Gentoo vs. andere	1373	54.1	330	25.368	<0.0001

Gentoo-Pinguine sind im Mittel um 1373 g schwerer als der Durchschnitt der beiden anderen Arten.

5.7.3 - Zweifaktorielle Varianzanalyse

Nun untersuchen wir, ob sich die Gewichtsunterschiede zwischen den Geschlechtern je nach Art unterscheiden. `aov_ez()` aus `afex` erwartet den Namen der ID-Variable, der abhängigen Variable, den Datensatz und die Faktoren. Faktoren, die zwischen Personen (hier: Tieren) variieren, übergibt man als `between`:

```
m_2f <- aov_ez(id = "id", dv = "masse_g", data = pinguine,
               between = c("art", "geschlecht"))
m_2f
```

Anova Table (Type 3 tests)

Response: masse_g

```

      Effect      df      MSE      F ges p.value
1      art 2, 327 95726.69 746.92 *** .820 <.001
2 geschlecht 1, 327 95726.69 311.84 *** .488 <.001
3 art:geschlecht 2, 327 95726.69 8.76 *** .051 <.001
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

```

Die Ausgabe zeigt für jeden Effekt die Freiheitsgrade, die mittlere Fehlerquadratsumme (MSE), den F-Wert, das generalisierte η^2 (ges) und den p-Wert. `afex` verwendet automatisch Effektkodierung und Typ-III-Quadratsummen. Zum Vergleich: Die Typ-I-Quadratsummen von `anova()` hängen bei ungleichen Zellbesetzungen von der Reihenfolge der Faktoren ab.

```
anova(lm(masse_g ~ art * geschlecht, data = pinguine))
```

Analysis of Variance Table

```

Response: masse_g
      Df    Sum Sq Mean Sq F value    Pr(>F)
art      2 145190219 72595110  758.358 < 2.2e-16 ***
geschlecht 1  37090262 37090262  387.460 < 2.2e-16 ***
art:geschlecht 2  1676557  838278   8.757 0.0001973 ***
Residuals  327  31302628    95727
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

```

```
anova(lm(masse_g ~ geschlecht * art, data = pinguine))
```

Analysis of Variance Table

```

Response: masse_g
      Df    Sum Sq Mean Sq F value    Pr(>F)
geschlecht 1  38878897 38878897 406.145 < 2.2e-16 ***
art      2 143401584 71700792 749.016 < 2.2e-16 ***
geschlecht:art 2  1676557  838278   8.757 0.0001973 ***
Residuals  327  31302628    95727
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

```

Die Interaktion ist signifikant. Um sie zu verstehen, bilden wir die Zellmittelwerte ab (Abbildung 5.5) und berechnen die **einfachen Effekte** des Geschlechts für jede Art. Der senkrechte Strich in der Formel `~ geschlecht | art` bedeutet „Geschlecht, getrennt für jede Art,“:

```

versatz <- position_dodge(width = 0.3)

ggplot(pinguine, aes(art, masse_g, colour = geschlecht,
group = geschlecht)) +
  stat_summary(fun = mean, geom = "line", position =
versatz) +
  stat_summary(fun.data = mean_se, fun.args = list(mult
= 1.96),
              geom = "pointrange", position = versatz) +
  scale_colour_manual(values = c(weiblich = blau, männlich

```

Da die Zellen hier fast gleich groß sind, sind die Unterschiede zwischen den Reihenfolgen klein. Bei stark unbalancierten Designs können sie erheblich sein. Der Test der Interaktion, die in beiden Formeln zuletzt kommt, ist identisch.

```
= akzent)) +
labs(x = NULL, y = "Körpergewicht (g)", colour = NULL) +
theme(legend.position = "top")
```

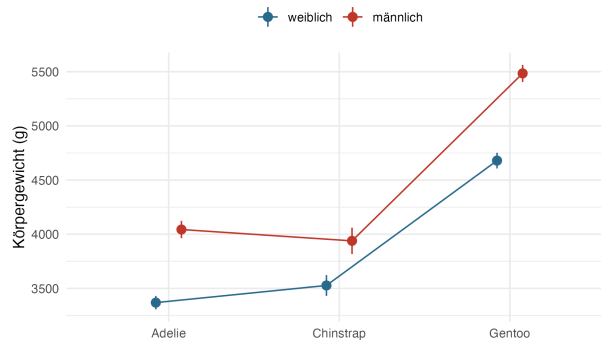


Abbildung 5.5: Mittleres Körpergewicht nach Art und Geschlecht mit 95%-Konfidenzintervallen. Die Linien verlaufen nicht parallel: Der Geschlechtsunterschied ist bei Gentoo-Pinguinen am größten.

```
emm_2f <- emmeans(m_2f, ~ geschlecht | art)
pairs(emm_2f)
```

```
art = Adelie:
contrast      estimate    SE  df t.ratio p.value
weiblich - männlich    -675  51.2 327  -13.174 <0.0001

art = Chinstrap:
contrast      estimate    SE  df t.ratio p.value
weiblich - männlich    -412  75.0 327   -5.487 <0.0001

art = Gentoo:
contrast      estimate    SE  df t.ratio p.value
weiblich - männlich    -805  56.7 327  -14.188 <0.0001
```

Männliche Tiere sind bei allen drei Arten schwerer, der Unterschied ist aber unterschiedlich groß – eine ordinale Interaktion. Ob sich der Geschlechtsunterschied zwischen zwei bestimmten Arten unterscheidet, prüfen **Interaktionskontraste**, also Differenzen von Differenzen:

```
contrast(emm_2f, interaction = "pairwise", by = NULL)
```

```
geschlecht_pairwise art_pairwise      estimate    SE  df
t.ratio p.value
weiblich - männlich Adelie - Chinstrap    -263  90.8 327
-2.894  0.0041
weiblich - männlich Adelie - Gentoo       130  76.4 327
1.706  0.0889
weiblich - männlich Chinstrap - Gentoo     393  94.1 327
4.181 <0.0001
```

5.7.4 - Moderation mit einem kontinuierlichen Prädiktor

Im Regressionskapitel hatten wir angenommen, dass der Zusammenhang zwischen Flossenlänge und Körpergewicht bei allen Arten gleich ist. Jetzt prüfen wir das mit einem Interaktionsterm. Die Flossenlänge zentrieren wir vorher, damit der Koeffizient der Art den Unterschied bei durchschnittlicher Flossenlänge beschreibt:

```
pinguine <- pinguine |>
  mutate(flosse_c = flosse_mm - mean(flosse_mm))

m_add <- lm(masse_g ~ flosse_c + art, data = pinguine)
m_mod <- lm(masse_g ~ flosse_c * art, data = pinguine)

anova(m_add, m_mod)
```

Analysis of Variance Table

```
Model 1: masse_g ~ flosse_c + art
Model 2: masse_g ~ flosse_c * art
  Res.Df    RSS Df Sum of Sq    F Pr(>F)
1     329 45843144
2     327 44391669  2   1451475 5.346 0.005193 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1
```

```
broom::tidy(m_mod)
```

```
# A tibble: 6 × 5
  term                estimate std.error statistic  p.value
<chr>                <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)        4061.         59.4      68.4 9.37e-196
2 flosse_c            32.7          4.69      4.69  6.97e-11
3 artChinstrap       -150.          81.1      -1.85  6.46e- 2
4 artGentoo           150.          108.       1.39  1.66e- 1
5 flosse_c:artChinstrap  1.88          7.86      0.240 8.11e- 1
6 flosse_c:artGentoo    21.5          6.97      3.08 2.23e- 3
```

Der Koeffizient `flosse_c` ist die Steigung in der Referenzgruppe (Adelie), die Koeffizienten `flosse_c:artChinstrap` und `flosse_c:artGentoo` sind die Unterschiede der Steigungen zu dieser Referenz. Die einfachen Steigungen für jede Art berechnet `emtrends()` direkt:

```
emtrends(m_mod, ~ art, var = "flosse_c")
```

```
art      flosse_c.trend SE df lower.CL upper.CL
Adelie    32.7 4.69 327    23.5    41.9
Chinstrap  34.6 6.31 327    22.2    47.0
Gentoo    54.2 5.15 327    44.0    64.3
```

`flosse_c * art` ist eine Abkürzung für `flosse_c + art + flosse_c:art`. Der Doppelpunkt bezeichnet den Produktterm allein.

Confidence level used: 0.95

Abbilden lässt sich das Modell mit `geom_smooth()`, das für jede Art eine eigene Gerade schätzt – genau das, was das Modell mit Interaktion tut (Abbildung 5.6):

```
ggplot(pinguine, aes(flosse_mm, masse_g, colour = art)) +
  geom_point(alpha = 0.3, size = 1) +
  geom_smooth(method = "lm", formula = y ~ x, se = TRUE) +
  scale_colour_manual(values = c(Adelie = blau, Chinstrap
                                = "grey45",
                                Gentoo = akzent)) +
  labs(x = "Flossenlänge (mm)", y = "Körpergewicht (g)",
       colour = NULL) +
  theme(legend.position = "top")
```

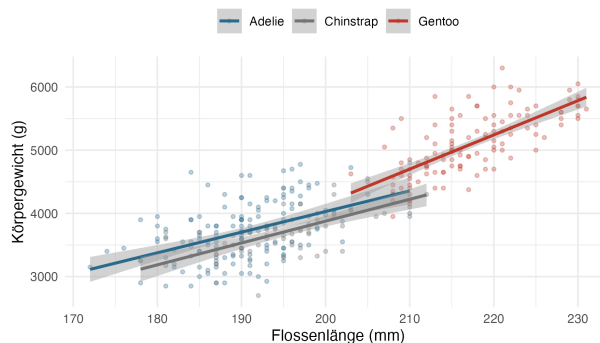


Abbildung 5.6: Körpergewicht in Abhängigkeit von der Flossenlänge, getrennt nach Art. Die Steigungen der Geraden (einfache Steigungen) unterscheiden sich zwischen den Arten.

5.7.5 - Eine Prä-post-Studie auswerten

Für die Auswertung eines Prä-post-Designs simulieren wir eine randomisierte Studie: 50 Beschäftigte nehmen an einem Resilienztraining teil, 50 bilden die Kontrollgruppe. Das Wohlbefinden wird vorher und nachher auf einer Skala mit Mittelwert 50 und Standardabweichung 10 gemessen. Die Prä-post-Korrelation beträgt $\rho = .6$, alle verbessern sich um 3 Punkte (etwa durch die Jahreszeit), und das Training bewirkt zusätzlich 5 Punkte. Der wahre Effekt entspricht also $d = 0,5$.

```
simuliere_prepost <- function(n_pro_gruppe = 50, rho =
0.6, effekt = 5, sd = 10) {
  n <- 2 * n_pro_gruppe
  gruppe <- rep(c("Kontrolle", "Training"), each =
n_pro_gruppe)
  prae <- rnorm(n, mean = 50, sd = sd)
  post <- 53 + rho * (prae - 50) + sqrt(1 - rho^2) *
rnorm(n, 0, sd) +
  effekt * (gruppe == "Training")
  tibble(id = 1:n,
         gruppe = factor(gruppe, levels = c("Kontrolle",
"Training")),
         prae, post)
```

```

}

set.seed(2026)
studie <- simulierte_prepost()

studie |>
  summarise(across(c(prae, post), list(m = mean, sd = sd)),
            r = cor(prae, post), .by = gruppe)

```

```

# A tibble: 2 × 6
  gruppe   prae_m prae_sd post_m post_sd     r
<fct>   <dbl>   <dbl> <dbl>   <dbl> <dbl>
1 Kontrolle  49.8   9.76   53.9   9.97 0.530
2 Training   48.2  10.3   57.9   9.00 0.650

```

Für die rmANOVA und für Abbildungen brauchen wir die Daten im **langen Format**, mit einer Zeile pro Person und Messzeitpunkt. `pivot_longer()` kennen Sie aus dem Kapitel zum Datenmanagement:

```

studie_lang <- studie |>
  pivot_longer(c(prae, post), names_to = "zeit", values_to =
    "wohlbefinden") |>
  mutate(zeit = factor(zeit, levels = c("prae", "post"),
    labels = c("vorher", "nachher")))

```

Wie immer bietet es sich an, sich die Daten erst einmal anzuschauen, bevor man anfängt komplexere Modelle an die Daten anzupassen.

```

versatz <- position_dodge(0.15)

ggplot(studie_lang, aes(zeit, wohlbefinden, colour =
  gruppe, group = gruppe)) +
  stat_summary(fun = mean, geom = "line", position =
    versatz) +
  stat_summary(fun.data = mean_cl_boot,
    geom = "pointrange", position = versatz) +
  scale_colour_manual(values = c(Kontrolle = blau,
    Training = akzent)) +
  labs(x = NULL, y = "Wohlbefinden", colour = NULL) +
  theme(legend.position = "top")

```

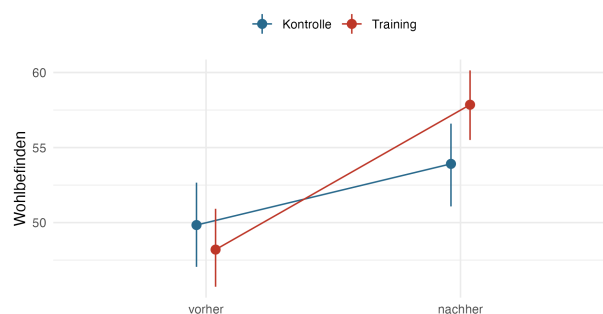


Abbildung 5.7: Mittleres Wohlbefinden vor und nach der Intervention mit 95%-Konfidenzintervallen. Beide Gruppen verbessern sich, die Trainingsgruppe stärker.

Nun rechnen wir die vier Auswertungsvarianten. Wir machen das nicht, weil Sie Selbst später alle Varianten rechnen sollen, sondern weil man hier nochmal sehr schön sehen kann, dass die ANCOVA für diese Art von Studien das beste Verfahren ist. Die ersten drei sind lineare Modelle; `I()` sorgt dafür, dass `R post - prae` als Rechenoperation versteht und nicht als Formelsyntax:

```
m_post <- lm(post ~ gruppe, data = studie)
m_diff <- lm(I(post - prae) ~ gruppe, data = studie)
m_ancova <- lm(post ~ prae + gruppe, data = studie)

m_rm <- aov_ez(id = "id", dv = "wohlbefinden", data =
  studie_lang,
  between = "gruppe", within = "zeit")
m_rm
```

Anova Table (Type 3 tests)

```
Response: wohlbefinden
      Effect    df    MSE      F ges p.value
1    gruppe 1, 98 151.49    0.44 .004    .510
2      zeit 1, 98  39.58 59.52 *** .112    <.001
3 gruppe:zeit 1, 98  39.58   9.82 ** .020    .002
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1
```

Die Effektschätzungen der drei linearen Modelle stellen wir nebeneinander:

```
pp_vergleich <- list("Nur Post" = m_post,
  "Veränderungswerte" = m_diff,
  "ANCOVA" = m_ancova) |>
  map(\(m) broom::tidy(m, conf.int = TRUE) |> filter(term
    == "gruppeTraining")) |>
  list_rbind(names_to = "methode") |>
  select(methode, estimate, std.error, conf.low,
    conf.high, statistic, p.value)

pp_vergleich
```

```
# A tibble: 3 × 7
  methode estimate std.error conf.low conf.high
statistic p.value
  <chr>      <dbl>    <dbl>    <dbl>    <dbl>
<dbl> <dbl>
1 Nur Post      3.94      1.90     0.172     7.71
2.08 0.0406
2 Veränderungswerte 5.58      1.78     2.04     9.11
3.13 0.00228
3 ANCOVA      4.85      1.55     1.77     7.92
3.13 0.00234
```

In allen Auswertungsvarianten ist der Effekt zwar signifikant, aber unterschiedlich groß. Außerdem ist Präzision unterschiedlich. In der Spalte `std.error` sehen Sie, dass die ANCOVA den Effekt am präzisesten schätzt. Außerdem können Sie prüfen, dass der F-Wert der Interaktion `gruppe:zeit` in der rmANOVA (9.82) genau das Quadrat des t-Werts der Veränderungswerte ist (9.82). Die Voraussetzung paral-

leler Geraden prüfen wir, indem wir die Interaktion von Prä-Wert und Gruppe aufnehmen, und die adjustierten Mittelwerte liefert wieder `emmeans()`:

```
anova(m_ancova, lm(post ~ prae * gruppe, data = studie))
```

Analysis of Variance Table

```
Model 1: post ~ prae + gruppe
Model 2: post ~ prae * gruppe
  Res.Df    RSS Df Sum of Sq    F Pr(>F)
1     97 5792.6
2     96 5791.1  1    1.5276 0.0253 0.8739
```

```
emmeans(m_ancova, ~ gruppe)
```

gruppe	emmean	SE	df	lower.CL	upper.CL
Kontrolle	53.5	1.09	97	51.3	55.6
Training	58.3	1.09	97	56.1	60.5

Confidence level used: 0.95

`emmeans()` setzt die Kovariate automatisch auf ihren Mittelwert. Die Differenz der adjustierten Mittelwerte ist exakt der Koeffizient `gruppeTraining` der ANCOVA.

5.7.6 - Effektstärken berechnen

Für die Effektstärke nach Gleichung (5.6) brauchen wir die Mittelwerte und Standardabweichungen beider Gruppen zu beiden Zeitpunkten:

```
kw <- studie |>
  summarise(n = n(), m_prae = mean(prae), m_post =
    mean(post),
            sd_prae = sd(prae), sd_post = sd(post), .by
            = gruppe)

gepoolt <- function(sd, n) sqrt(sum((n - 1) * sd^2) /
  (sum(n) - 2))

sd_prae_gep <- gepoolt(kw$sd_prae, kw$n)
sd_post_gep <- gepoolt(kw$sd_post, kw$n)
c_p <- 1 - 3 / (4 * (sum(kw$n) - 2) - 1)

veraenderung <- kw$m_post - kw$m_prae      # Kontrolle,
Training
d_ppc <- c_p * (veraenderung[2] - veraenderung[1]) /
sd_prae_gep

d_ancova <- coef(m_ancova)["gruppeTraining"] /
sd_post_gep

# zum Vergleich: bezogen auf die Streuung der
Veränderungswerte
sd_diff_gep <- studie |>
  summarise(sd = sd(post - prae), n = n(), .by = gruppe)
|>
  with(gepoolt(sd, n))
d_veraenderung <- coef(m_diff)["gruppeTraining"] /
sd_diff_gep
```

```
round(c(d_ppc = d_ppc, d_ancova = unname(d_ancova),
        d_veraenderung = unname(d_veraenderung)), 2)
```

d_ppc	d_ancova	d_veraenderung
0.55	0.51	0.63

Die beiden empfohlenen Maße d_{ppc} und ANCOVA liegen nahe am wahren Wert von 0,5. Das auf die Veränderungswerte bezogene d fällt deutlich größer aus, obwohl es denselben Effekt beschreiben sollte. In der Praxis sollten Sie daher die ANCOVA verwenden.

5.7.7 - Warum die ANCOVA gewinnt: eine Simulation

Eine einzelne Studie kann zufällig für die eine oder andere Methode günstig ausfallen. Ob die theoretischen Überlegungen zur Präzision stimmen, prüfen wir mit einer kleinen Simulationsstudie. Wir wiederholen die Studie 1000-mal, werten jede Wiederholung mit den drei Methoden aus und vergleichen die Verteilung der Schätzungen und den Anteil signifikanter Ergebnisse, also die empirische Teststärke:

```
werte_aus <- function(d) {
  list("Nur Post" = lm(post ~ gruppe, data = d),
       "Veränderungswerte" = lm(I(post - prae) ~ gruppe,
data = d),
       "ANCOVA" = lm(post ~ prae + gruppe, data
= d)) |>
  map(\(m) broom::tidy(m) |> filter(term ==
"gruppeTraining")) |>
  list_rbind(names_to = "methode")
}

set.seed(123)
sim <- map(1:1000, \(i) werte_aus(simuliere_prepost()))
|>
  list_rbind(names_to = "durchgang") |>
  mutate(methode = fct_inorder(methode))

sim_ergebnis <- sim |>
  summarise(mittlere_schaetzung = mean(estimate),
            sd_schaetzung = sd(estimate),
            teststaerke = mean(p.value < .05),
            .by = methode)
sim_ergebnis
```

```
# A tibble: 3 × 4
  methode      mittlere_schaetzung sd_schaetzung
<fct>          <dbl>          <dbl>
1 Nur Post      5.13            1.87
2 Veränderungswerte 5.06            1.79
3 ANCOVA        5.09            1.55
```

```
ggplot(sim, aes(estimate, colour = methode)) +
  geom_vline(xintercept = 5, linetype = "dashed", colour
= "grey40") +
  geom_density(linewidth = 1) +
```

```
scale_colour_manual(values = c(`Nur Post` = "grey50",
                               `Veränderungswerte`
                               = blau,
                               ANCOVA = akzent)) +
  labs(x = "Geschätzter Trainingseffekt", y = "Dichte",
       colour = NULL) +
  theme(legend.position = "top")
```

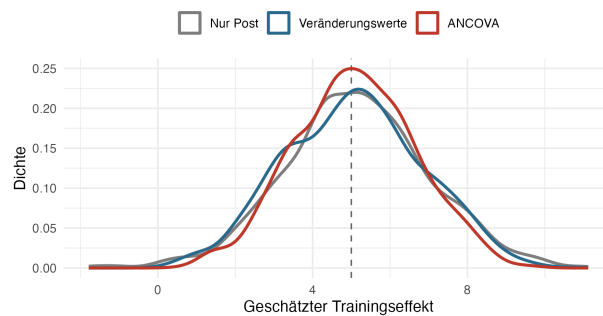


Abbildung 5.8: Verteilung der geschätzten Trainingseffekte in 1000 simulierten Studien ($n = 50$ pro Gruppe, $\rho = .6$). Alle drei Methoden sind im Mittel unverzerrt (wahrer Effekt: gestrichelte Linie), die ANCOVA streut aber am wenigsten.

Alle drei Methoden treffen im Mittel den wahren Effekt von 5 Punkten. Die Teststärke unterscheidet sich aber erheblich: Mit derselben Stichprobe entdeckt die ANCOVA den Effekt in 88 % der Studien, der reine Post-Vergleich nur in 74 %.

5.7.8 - Messwiederholung mit mehr als zwei Zeitpunkten

Bei mehr als zwei Messzeitpunkten spielt die sog. Sphärizität eine Rolle. Als Beispiel verwenden wir den Datensatz `obk.long`, der mit `afex` geladen wird. Er enthält die Daten eines fiktiven Experiments von O'Brien und Kaiser mit drei Behandlungsgruppen (`treatment`: `control`, `A`, `B`), die vor der Behandlung (`pre`), danach (`post`) und in einer Nachuntersuchung (`fup`) gemessen wurden. Zu jedem Zeitpunkt werden fünf Messungen durchgeführt, die wir mit `fun_aggregate = mean` zu einem Mittelwert zusammenfassen:

```
data(obk.long, package = "afex")

m_obk <- aov_ez(id = "id", dv = "value", data = obk.long,
               between = "treatment", within = "phase",
               fun_aggregate = mean)

m_obk
```

Anova Table (Type 3 tests)

Response: value						
	Effect	df	MSE	F	ges	p.value
1	treatment	2, 13	6.41	2.91	+	.269 .090
2	phase	1.74, 22.64	0.81	19.29	***	.212 <.001

```
3 treatment:phase 3.48, 22.64 0.81 5.43 ** .131 .004
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
' ' 1

Sphericity correction method: GG
```

In der Zeile `phase` und `treatment:phase` sehen Sie Freiheitsgrade mit Nachkommastellen: Das ist die Greenhouse-Geisser-Korrektur, die `afex` standardmäßig anwendet. Die unkorrigierten Werte und die Sphärizitätstests zeigt `summary(m_obk)`. Einfache Effekte und Post-hoc-Tests funktionieren genau wie bei Designs ohne Messwiederholung:

```
emm_obk <- emmeans(m_obk, ~ phase | treatment)
pairs(emm_obk, adjust = "holm")
```

```
treatment = control:
contrast estimate SE df t.ratio p.value
fup - post 0.400 0.486 13 0.822 1.0000
fup - pre 0.200 0.471 13 0.425 1.0000
post - pre -0.200 0.627 13 -0.319 1.0000

treatment = A:
contrast estimate SE df t.ratio p.value
fup - post 0.750 0.544 13 1.379 0.1911
fup - pre 2.250 0.526 13 4.275 0.0027
post - pre 1.500 0.700 13 2.141 0.1035

treatment = B:
contrast estimate SE df t.ratio p.value
fup - post 0.714 0.411 13 1.738 0.1059
fup - pre 3.143 0.398 13 7.899 <0.0001
post - pre 2.429 0.530 13 4.586 0.0010

P value adjustment: holm method for 3 tests
```

5.7.9 - Fallzahlberechnung

Das Paket `pwr` berechnet Fallzahlen für die Standardfälle. Lässt man eines der Argumente `n`, `d` (bzw. `f`), `sig.level` oder `power` weg, wird genau dieses berechnet. Für einen t-Test bzw. den reinen Post-Vergleich mit $d = 0,5$:

```
library(pwr)

plan_t <- pwr.t.test(d = 0.5, sig.level = 0.05, power
= 0.80)
plan_t
```

Two-sample t test power calculation

```
n = 63.76561
d = 0.5
sig.level = 0.05
power = 0.8
alternative = two.sided
```

NOTE: n is number in *each* group

Für die ANCOVA multipliziert man die Fallzahl mit $1 - \rho^2$. Bei einer erwarteten Prä-post-Korrelation von .6 ergibt das:

```
n_ancova <- ceiling(plan_t$n * (1 - 0.6^2))
n_ancova
```

```
[1] 41
```

Für eine einfaktorielle ANOVA verwendet `pwr.anova.test()` die Effektgröße f nach Cohen, die bei zwei Gruppen $d/2$ entspricht. Für drei Gruppen und einen mittleren Effekt von $f = 0,25$:

```
pwr.anova.test(k = 3, f = 0.25, sig.level = 0.05, power
= 0.80)
```

```
Balanced one-way analysis of variance power
calculation
```

```
      k = 3
      n = 52.3966
      f = 0.25
sig.level = 0.05
power = 0.8
```

NOTE: n is number in each group

Ob die Formel für die ANCOVA stimmt, können wir mit derselben Simulationslogik wie oben prüfen. Wir simulieren Studien mit 41 Personen pro Gruppe und bestimmen die Teststärke der ANCOVA und des reinen Post-Vergleichs:

```
set.seed(99)
pwr_sim <- map(1:1000, \ (i)
werte_aus(simuliere_prepost(n_pro_gruppe = n_ancova))) |>
  list_rbind() |>
  summarise(teststaerke = mean(p.value < .05), .by =
methode)
pwr_sim
```

```
# A tibble: 3 × 2
  methode      teststaerke
  <chr>      <dbl>
1 Nur Post      0.604
2 Veränderungswerte 0.685
3 ANCOVA        0.784
```

Die ANCOVA erreicht mit 41 statt 64 Personen pro Gruppe die geplanten 80 %, der Post-Vergleich mit derselben Stichprobe nicht. Diese Art der simulationsbasierten Fallzahlplanung lässt sich auf jedes Design übertragen, für das es keine Formel gibt: Man ersetzt einfach die Funktionen `simuliere_prepost()` und `werte_aus()` durch das geplante Design und die geplante Auswertung.

Die Werte von `n` gelten in `pwr` immer **pro Gruppe**. Runden Sie stets auf.

5.8 - Fazit

- Die **Varianzanalyse** zerlegt die Gesamtvarianz in einen Anteil zwischen und einen innerhalb der Gruppen und testet mit dem F-Test, ob sich Mittelwerte unterscheiden. Sie ist ein Spezialfall des Allgemeinen Linearen Modells.
- Nach einem signifikanten F-Test klären **geplante Kontraste** oder **Post-hoc-Tests** mit Korrektur für multiples Testen, welche Gruppen sich unterscheiden. Als Effektstärken dienen η^2 , ω^2 und das partielle bzw. generalisierte η^2 .
- **Interaktionen** sind Produktterme im linearen Modell und beschreiben, ob und wie der Effekt eines Faktors von einem anderen abhängt. Sind Interaktionen im Modell, hängt die Bedeutung der Haupteffekte von der Kodierung ab, und bei unbalancierten Designs vom Typ der Quadratsummen. Tests der Interaktionen benötigen viel größere Stichproben als Haupteffekte.
- **Moderation** ist eine Interaktion mit einem kontinuierlichen Prädiktor. Man zentriert die Prädiktoren und interpretiert die Interaktion über einfache Steigungen und Abbildungen.
- Die **ANCOVA** erhöht in randomisierten Studien die Teststärke, weil die Kovariate Fehlervarianz aufklären kann. In nicht randomisierten Studien kann sie Gruppenunterschiede nur begrenzt „herausrechnen“.
- Die **rmANOVA** entfernt stabile Personenunterschiede aus dem Fehlerterm. Bei mehr als zwei Messzeitpunkten korrigiert man für Verletzungen der Sphärizität mit Greenhouse-Geisser.
- In randomisierten **Prä-post-Designs** ist die ANCOVA mit dem Prä-Wert als Kovariate die präziseste Auswertung. Effektstärken bezieht man auf die Standardabweichung der Rohwerte, nicht der Veränderungswerte, und getrennte Tests in beiden Gruppen ersetzen nie den Test des Gruppenunterschieds.
- Die **Fallzahlplanung** gehört vor die Datenerhebung. Sie sollte auf dem kleinsten relevanten Effekt und dem tatsächlich geplanten Auswertungsmodell beruhen; für komplexe Designs ist die Simulation das Mittel der Wahl.
- In R rechnet man Varianzanalysen mit `lm()` oder `afex::aov_ez()`, berechnet adjustierte Mittelwerte, Kontraste und einfache Effekte mit `emmeans`, Effektstärken mit `effectsize` und Fallzahlen mit `pwr` oder per Simulation.

5.8.1 - Vertiefung

Das Standardwerk zur Planung und Auswertung von Experimenten ist *Designing Experiments and Analyzing Data* von Maxwell, Delaney und Kelley (2018), das konsequent aus der Perspektive von Modellvergleichen argumentiert und alle Themen dieses Kapitels ausführlich behandelt. Zur Auswertung von Prä-post-Studien sind der kurze und sehr klare Beitrag von Vickers und Altman (2001) und die etwas technischere Arbeit von Van Breukelen (2006) zu empfehlen, die auch den Unterschied zwischen randomisierten und nicht randomisierten Studien herausarbeitet. Effektstärken für Prä-post-Designs behandelt Morris (2008). Wie man eine Fallzahlen mit und ohne Poweranalysen sinnvoll begründet, beschreibt Lakens (2022). Und wer Interaktionen mit kontinuierlichen Prädiktoren vertiefen möchte, findet bei Aiken und West (1991) den Klassiker zum Thema.

5.8.2 - Referenzen

6. Logistische Regression

7. Das LASSO und andere Shrinkage-Verfahren

8. Mehrebenenmodelle

III

Faktorenanalysen und Strukturgleichungsmodelle

9	Exploratorische Faktorenanalyse .	133
10	Konfirmatorische Faktorenanalyse .	135
11	Messinvarianz	137
12	Veränderung modellieren	139

9. Exploratorische Faktorenanalyse

10. Konfirmatorische Faktorenanalyse

11. Messinvarianz

12. Veränderung modellieren

IV Einzelne Verfahren

13	Conjoint-Analyse	143
13	References	145

13. Conjoint-Analyse

References

- Aiken, L. S., & West, S. G. (1991). *Multiple Regression: Testing and Interpreting Interactions*. Sage.
- Albers, C., & Lakens, D. (2018). When Power Analyses Based on Pilot Data Are Biased: Inaccurate Effect Size Estimators and Follow-up Bias. *Journal of Experimental Social Psychology*, 74, 187–195. <https://doi.org/10.1016/j.jesp.2017.09.004>
- Anscombe, F. J. count. (1973). Graphs in statistical analysis. *The American Statistician*, 27(1), 17–21. <https://doi.org/10.1080/00031305.1973.10478966>
- Arel-Bundock, V. (2022). modelsummary: Data and Model Summaries in R. *Journal of Statistical Software*, 103(1), 1–23. <https://doi.org/10.18637/jss.v103.i01>
- Beerendonk, L., Mejías, J. F., Nuiten, S. A., Gee, J. W. de, Fahrenfort, J. J., & Gaal, S. van. (2024). A disinhibitory circuit mechanism explains a general principle of peak performance during mid-level arousal. *Proceedings of the National Academy of Sciences*, 121(5). <https://doi.org/10.1073/pnas.2312898121>
- Begley, C. G., & Ellis, L. M. (2012). Raise standards for preclinical cancer research. *Nature*, 483(7391), 531–533. <https://doi.org/10.1038/483531a>
- Bem, D. J. (2011). Feeling the future: Experimental evidence for anomalous retroactive influences on cognition and affect. *Journal of Personality and Social Psychology*, 100(3), 407–425. <https://doi.org/10.1037/a0021524>
- Ben-Shachar, M. S., Lüdtke, D., & Makowski, D. (2020). effectsize: Estimation of Effect Size Indices and Standardized Parameters. *Journal of Open Source Software*, 5(56), 2815. <https://doi.org/10.21105/joss.02815>
- Bland, J. M., & Altman, D. G. (2011). Comparisons against Baseline within Randomised Groups Are Often Used and Can Be Highly Misleading. *Trials*, 12, 264. <https://doi.org/10.1186/1745-6215-12-264>
- Borm, G. F., Fransen, J., & Lemmens, W. A. (2007). A simple sample size formula for analysis of covariance in randomized clinical trials. *Journal of clinical epidemiology*, 60(12), 1234–1238. <https://doi.org/10.1016/j.jclinepi.2007.02.006>
- Bortz, J., & Schuster, C. (2010). *Statistik für Human- und Sozialwissenschaftler* (7. Aufl.). Springer.
- Box, G. E. P. (1966). Use and Abuse of Regression. *Technometrics*, 8(4), 625. <https://doi.org/10.2307/1266635>
- Box, G. E. P. (1976). Science and statistics. *Journal of the American Statistical Association*, 71(356), 791–799.
- Brown, N. J. L., & Heathers, J. A. J. (2016). The GRIM Test: A Simple Technique Detects Numerous Anomalies in the Reporting of Results in Psychology. *Social Psychological and Personality Science*, 8(4), 363–369. <https://doi.org/10.1177/1948550616673876>
- Cairo, A. (2012). *The Functional Art: An introduction to information graphics and visualization*. New Riders.
- Cleveland, W. S., & McGill, R. (1984). Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods. *Journal of the American Statistical Association*, 79(387), 531–554. <https://doi.org/10.1080/01621459.1984.10478080>

- Cohen, J. (1968). Multiple regression as a general data-analytic system. *Psychological Bulletin*, 70(6p1), 426.
- Cohen, J. (1988). *Statistical power analysis for the behavioural sciences* (2nd Aufl.). Lawrence Erlbaum Associates.
- Cohen, J. (1990). Things I Have Learned (So Far). *American Psychologist*, 45(12), 1304–1312. <https://doi.org/10.1037/0003-066X.45.12.1304>
- Cohen, J., Cohen, P., West, S. G., & Aiken, L. S. (2003). *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences* (3. Aufl.). Lawrence Erlbaum.
- Corti, L. (2020). *Managing and sharing research data: A guide to good practice* (V. V. den Eynden, L. Bishop, M. Woollard, M. Haaker, & S. Summers, Hrsg.; 2nd edition). SAGE.
- Cumming, G., & Calin-Jageman, R. (2017). *Introduction to the New Statistics: Estimation, Open Science, and Beyond*. Routledge.
- Cumming, Geoff. (2012). *Understanding the new statistics: Effect sizes, confidence intervals, and meta-analysis*. Routledge.
- Cumming, G., & Finch, S. (2005). Inference by Eye: Confidence Intervals and How to Read Pictures of Data. *American Psychologist*, 60(2), 170–180. <https://doi.org/10.1037/0003-066X.60.2.170>
- Curran, P. G. (2016). Methods for the detection of carelessly invalid responses in survey data. *Journal of Experimental Social Psychology*, 66, 4–19. <https://doi.org/10.1016/j.jesp.2015.07.006>
- Döring, N. (2023). *Forschungsmethoden und Evaluation: für Human- und Sozialwissenschaftler*. Springer.
- Eid, M., Gollwitzer, M., & Schmitt, M. (2017). *Statistik und Forschungsmethoden*.
- Faul, F., Erdfelder, E., G., L. A., & Buchner, A. (2007). G*Power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior Research Methods*, 39, 175–191.
- Field, A. (2026). *Discovering statistics using R* (2nd edition). Sage.
- Fisher, R. A. (1922). On the Mathematical Foundations of Theoretical Statistics. *Philosophical Transactions of the Royal Society of London. Series A*, 222, 309–368. <https://doi.org/10.1098/rsta.1922.0009>
- Fisher, R. A. (1925). *Statistical Methods for Research Workers*. Oliver; Boyd.
- Fisher, R. A. (1938). Presidential Address to the First Indian Statistical Congress. *Sankhyā*, 4(1), 14–17.
- Fox, J. (2016). *Applied Regression Analysis and Generalized Linear Models* (3. Aufl.). Sage.
- Galilei, G. (2016). *Sidereus Nuncius, or the sidereal messenger*. University of Chicago Press. (Original work published 1610)
- Galton, F. (1886). Regression Towards Mediocrity in Hereditary Stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, 15, 246–263. <https://doi.org/10.2307/2841583>
- Gelman, A. (2007). Scaling regression inputs by dividing by two standard deviations. *Statistics in Medicine*, 27(15), 2865–2873. <https://doi.org/10.1002/sim.3107>
- Gelman, A., Hill, J., & Vehtari, A. (2020). *Regression and Other Stories*. Cambridge University Press. <https://doi.org/10.1017/9781139161879>
- Gelman, A., & Loken, E. (2014). The Statistical Crisis in Science. *American Scientist*, 102(6), 460. <https://doi.org/10.1511/2014.111.460>
- Gelman, A., & Stern, H. (2006). The difference between significant and not significant is not itself statistically significant. *American Statistician*, 60(4), 328–331.
- Greenhouse, S. W., & Geisser, S. (1959). On Methods in the Analysis of Profile Data. *Psychometrika*, 24(2), 95–112. <https://doi.org/10.1007/BF02289823>

- Herndon, T., Ash, M., & Pollin, R. (2013). Does high public debt consistently stifle economic growth? A critique of Reinhart and Rogoff. *Cambridge Journal of Economics*, 38(2), 257–279. <https://doi.org/10.1093/cje/bet075>
- Hunsley, J., & Meyer, G. J. (2003). The Incremental Validity of Psychological Testing and Assessment: Conceptual, Methodological, and Statistical Issues. *Psychological Assessment*, 15(4), 446–455. <https://doi.org/10.1037/1040-3590.15.4.446>
- Johnson, J. W., & LeBreton, J. M. (2004). History and Use of Relative Importance Indices in Organizational Research. *Organizational Research Methods*, 7(3), 238–257. <https://doi.org/10.1177/1094428104266510>
- Lakens, D. (2013). Calculating and reporting effect sizes to facilitate cumulative science: a practical primer for t-tests and ANOVAs. *Frontiers in psychology*, 4, 863.
- Lakens, D. (2022). Sample Size Justification. *Collabra: Psychology*, 8(1), 33267. <https://doi.org/10.1525/collabra.33267>
- Lehr, R. (1992). Sixteen S-Squared over D-Squared: A Relation for Crude Sample Size Estimates. *Statistics in Medicine*, 11(8), 1099–1102. <https://doi.org/10.1002/sim.4780110811>
- Lindeløv, J. K. (2019). *Common Statistical Tests Are Linear Models (or: How to Teach Stats)*. <https://lindeloev.github.io/tests-as-linear/>.
- Long, J. S., & Ervin, L. H. (2000). Using Heteroscedasticity Consistent Standard Errors in the Linear Regression Model. *The American Statistician*, 54(3), 217–224. <https://doi.org/10.1080/00031305.2000.10474549>
- Maslow, A. (1962). *Toward a psychology of being*. D Van Nostrand. <https://doi.org/10.1037/10793-000>
- Matejka, J., & Fitzmaurice, G. (2017). Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 1290–1294. <https://doi.org/10.1145/3025453.3025912>
- Maxwell, S. E., Delaney, H. D., & Kelley, K. (2018). *Designing Experiments and Analyzing Data: A Model Comparison Perspective* (3. Aufl.). Routledge. <https://doi.org/10.4324/9781315642956>
- Meade, A. W., & Craig, S. B. (2012). Identifying careless responses in survey data. *Psychological Methods*, 17(3), 437–455. <https://doi.org/10.1037/a0028085>
- Meyer, G. J., Finn, S. E., Eyde, L. D., Kay, G. G., Moreland, K. L., Dies, R. R., Eisman, E. J., Kubiszyn, T. W., & Reed, G. M. (2001). Psychological testing and psychological assessment: A review of evidence and issues. *American Psychologist*, 56(2), 128–165. <https://doi.org/10.1037/0003-066x.56.2.128>
- Miller, G. A., & Chapman, J. P. (2001). Misunderstanding Analysis of Covariance. *Journal of Abnormal Psychology*, 110(1), 40–48. <https://doi.org/10.1037/0021-843X.110.1.40>
- Moosbrugger, H., & Kelava, A. (Hrsg.). (2020). *Testtheorie und Fragebogenkonstruktion*. Springer Berlin Heidelberg. <https://doi.org/10.1007/978-3-662-61532-4>
- Morris, S. B. (2008). Estimating Effect Sizes from Pretest-Posttest-Control Group Designs. *Organizational Research Methods*, 11(2), 364–386. <https://doi.org/10.1177/1094428106291059>
- Morris, T. P., White, I. R., & Crowther, M. J. (2019). Using simulation studies to evaluate statistical methods. *Statistics in Medicine*, 38(11), 2074–2102. <https://doi.org/10.1002/sim.8086>
- Munafò, M. R. et al. (2017). A manifesto for reproducible science. *Nature Human Behaviour*, 1, 0021.

- Nosek, B. A., Ebersole, C. R., DeHaven, A. C., & Mellor, D. T. (2018). The preregistration revolution. *Proceedings of the National Academy of Sciences*, 115(11), 2600–2606. <https://doi.org/10.1073/pnas.1708274114>
- Open Science Collaboration. (2015). Estimating the reproducibility of psychological science. *Science*, 349(6251). <https://doi.org/10.1126/science.aac4716>
- Orben, A., & Przybylski, A. K. (2019). The association between adolescent well-being and digital technology use. *Nature Human Behaviour*, 3(2), 173–182. <https://doi.org/10.1038/s41562-018-0506-1>
- Popper, K. R. (2007). *Conjectures and refutations: The growth of scientific knowledge* (Repr.). Routledge.
- R Core Team. (2026). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. <https://doi.org/10.32614/R.manuals>
- Reinhart, C. M., & Rogoff, K. S. (2010). Growth in a Time of Debt. *American Economic Review*, 100(2), 573–578. <https://doi.org/10.1257/aer.100.2.573>
- Sackett, P. R., Zhang, C., Berry, C. M., & Lievens, F. (2022). Revisiting Meta-Analytic Estimates of Validity in Personnel Selection: Addressing Systematic Overcorrection for Restriction of Range. *Journal of Applied Psychology*, 107(11), 2040–2068. <https://doi.org/10.1037/apl0000994>
- Schmidt, F. L., & Hunter, J. E. (1998). The Validity and Utility of Selection Methods in Personnel Psychology: Practical and Theoretical Implications of 85 Years of Research Findings. *Psychological Bulletin*, 124(2), 262–274. <https://doi.org/10.1037/0033-2909.124.2.262>
- Sedlmeier, P., & Gigerenzer, G. (1989). Do studies of statistical power have an effect on the power of studies? *Psychological Bulletin*, 105(2), 309–316.
- Senn, S. (2006). Change from Baseline and Analysis of Covariance Revisited. *Statistics in Medicine*, 25(24), 4334–4344. <https://doi.org/10.1002/sim.2682>
- Shmueli, G. (2010). To explain or to predict? *Statistical Science*, 25(3), 289–310. <https://doi.org/10.1214/10-sts330>
- Shu, L. L., Mazar, N., Gino, F., Ariely, D., & Bazerman, M. H. (2012). RETRACTED: Signing at the beginning makes ethics salient and decreases dishonest self-reports in comparison to signing at the end. *Proceedings of the National Academy of Sciences*, 109(38), 15197–15200. <https://doi.org/10.1073/pnas.1209746109>
- Simmons, J. P., Nelson, L. D., & Simonsohn, U. (2011). False-positive psychology: Undisclosed flexibility in data collection and analysis allows presenting anything as significant. *Psychological science*, 22(11), 1359–1366.
- Tufte, E. R. (1983). *The visual display of quantitative information*. Graphics press.
- Tufte, E. R. (2006). Beautiful Evidence. In *Connecticut: Graphics Press LLC*. Graphics press Cheshire, CT.
- Tufte, E. R. (Hrsg.). (2013). *Envisioning information* (14. print.). Graphics Press.
- Tukey, J. W. (1962). The Future of Data Analysis. *The Annals of Mathematical Statistics*, 33(1), 1–67. <https://doi.org/10.1214/aoms/1177704711>
- Tukey, J. W. (1977). *Exploratory data analysis*. Reading, Mass.
- Van Breukelen, G. J. P. (2006). ANCOVA versus Change from Baseline Had More Power in Randomized Studies and More Bias in Nonrandomized Studies. *Journal of Clinical Epidemiology*, 59(9), 920–925. <https://doi.org/10.1016/j.jclinepi.2006.02.007>

- Vickers, A. J., & Altman, D. G. (2001). Analysing Controlled Trials with Baseline and Follow up Measurements. *BMJ*, 323(7321), 1123–1124. <https://doi.org/10.1136/bmj.323.7321.1123>
- Wagenmakers, E.-J., Marsman, M., Jamil, T., Ly, A., Verhagen, J., Love, J., Selker, R., Gronau, Q. F., Šmíra, M., Epskamp, S., Matzke, D., Rouder, J. N., & Morey, R. D. (2017). Bayesian inference for psychology. Part I: Theoretical advantages and practical ramifications. *Psychonomic Bulletin & Review*, 25(1), 35–57. <https://doi.org/10.3758/s13423-017-1343-3>
- Westfall, J., & Yarkoni, T. (2016). Statistically Controlling for Confounding Constructs Is Harder than You Think. *PLoS ONE*, 11(3), e0152719. <https://doi.org/10.1371/journal.pone.0152719>
- Wickham, H. (2010). A layered grammar of graphics. *Journal of Computational and Graphical Statistics*, 19(1), 3–28. <https://doi.org/10.1198/jcgs.2009.07098>
- Wickham, H. (2014). Tidy data. *Journal of Statistical Software*, 59. <https://doi.org/10.18637/jss.v059.i10>
- Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis* (2. Aufl.). Springer. <https://doi.org/10.1007/978-3-319-24277-4>
- Wilkinson, L. (2012). The Grammar of Graphics. In *Handbook of Computational Statistics*. Springer. https://doi.org/10.1007/978-3-642-21551-3_13
- Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., ... Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3(1). <https://doi.org/10.1038/sdata.2016.18>
- Williams, M. N., Grajales, C. A. G., & Kurkiewicz, D. (2013). Assumptions of Multiple Regression: Correcting Two Misconceptions. *Practical Assessment, Research, and Evaluation*, 18(11). <https://doi.org/10.7275/55hn-wk47>

Anhang

A. Erste Schritte mit R und RStudio

Der zweite Teil dieser Einführung ist praktischer Natur. Wir arbeiten in diesem Kurs durchgängig mit **R** – einer freien, quelloffenen Sprache für Statistik. Alle Analysen, Abbildungen und sogar dieses Buch selbst sind mit R (und dem Publikationssystem Quarto) erstellt. Bevor wir uns mit einzelnen Verfahren beschäftigen, richten wir daher Ihre Arbeitsumgebung ein und schauen uns die ersten Befehle an. Das **Datenmanagement** – also das Einlesen und Aufbereiten echter Datensätze – besprechen wir bewusst erst in einem späteren Kapitel. Hier geht es nur ums Ankommen.

A.1 - Was ist R und R Studio

R ist eine freie, quelloffene Programmiersprache, die speziell für Statistik, Datenanalyse und die Erstellung von Abbildungen entwickelt wurde (R Core Team, 2026). Anders als bei Programmen mit Menüs und Schaltflächen – etwa SPSS oder Excel – schreiben Sie in R Ihre Analysen als Befehle auf. Das wirkt am Anfang ungewohnt, hat aber einen großen Vorteil: Jeder Schritt ist schriftlich festgehalten, nachvollziehbar und jederzeit reproduzierbar. Sie können eine Auswertung Monate später exakt wiederholen oder mit einem Klick auf neue Daten anwenden. Genau diese Nachvollziehbarkeit ist in der wissenschaftlichen Arbeit unverzichtbar. Hinzu kommt, dass R kostenlos ist und durch tausende frei verfügbare Erweiterungen (Pakete) praktisch jedes moderne Verfahren beherrscht.

Ein weiterer Vorteil ist, dass Sprachmodelle wie Mistral, Claude oder ChatGPT hervorragend darin sind zu programmieren und Ihnen dadurch oft gut weiterhelfen können.

RStudio ist dagegen keine eigene Statistik-Software, sondern eine komfortable Arbeitsumgebung um R herum – eine sogenannte IDE (Integrated Development Environment). RStudio bündelt Editor, Konsole, Abbildungen und Hilfeseiten in einem Fenster und nimmt Ihnen viel Schreibarbeit ab. Beide zusammen bilden das Werkzeug, mit dem wir in diesem Kurs durchgängig arbeiten werden.

Hinweis

Man kann sich R als den Motor vorstellen, der im Hintergrund rechnet, und RStudio als das übersichtliche Cockpit, von dem aus Sie ihn bedienen.

A.2 - R und RStudio installieren

Zwei Dinge sind zu installieren, und die Reihenfolge ist wichtig: **erst R, dann RStudio**.

- **R** ist die eigentliche Statistik-Software – der Motor im Hintergrund, der rechnet.

- **RStudio** ist eine komfortable Arbeitsumgebung (eine sogenannte IDE) um R herum, mit Editor, Konsole und vielen Hilfen. RStudio funktioniert nur, wenn R bereits installiert ist.

Schritt 1 – R installieren. Öffnen Sie die Seite des *Comprehensive R Archive Network* (CRAN) unter <https://cran.r-project.org>, wählen Sie Ihr Betriebssystem (Windows, macOS oder Linux) und installieren Sie die aktuelle Version mit den Standardeinstellungen.

Schritt 2 – RStudio installieren. Laden Sie anschließend *RStudio Desktop* (Open-Source-Edition, kostenlos) von Posit herunter: <https://posit.co/download/rstudio-desktop/>. Auch hier reichen die Standardeinstellungen.

💡 Tipp

Wenn Sie beim Start von RStudio in der Konsole eine Zeile wie `R version 4.x.x` sehen, hat RStudio Ihr R gefunden und alles ist bereit.

A.3 - Die RStudio-Oberfläche

Nach dem ersten Start sehen Sie im Wesentlichen drei Bereiche (Panes):

- **Konsole** (links): Hier tippen Sie die R-Befehle direkt ein und bekommen sofort ein Ergebnis. Hier fangen wir gleich an.
- **Environment / History** (oben rechts): Hier sehen Sie, welche Objekte (Variablen, Datensätze) aktuell im Speicher liegen.
- **Files / Plots / Packages / Help** (unten rechts): Dateibrowser, erzeugte Abbildungen, installierte Pakete und die Hilfeseiten.

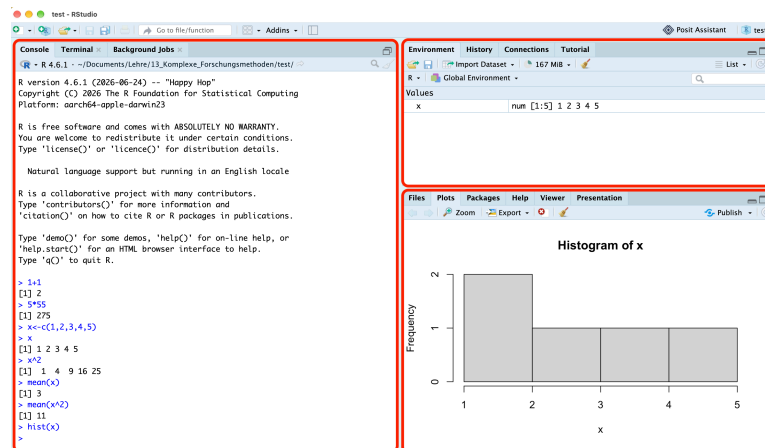


Abbildung A.1: Screenshot der Rstudio-Oberfläche

A.4 - R als Taschenrechner

Am einfachsten fangen Sie links in der **Konsole** an. Dort steht ein Prompt-Zeichen `>` und blinkt ein Cursor. Alles, was Sie eintippen und mit Enter bestätigen, wird sofort ausgewertet. R ist zunächst nichts anderes als ein sehr guter Taschenrechner:

```
[1] 5
```

Das [1] vor dem Ergebnis ist übrigens kein Tippfehler, sondern eine Positionsangabe – R nummeriert seine Ausgaben. Bei einem einzelnen Wert steht dort immer [1].

Die üblichen Rechenoperationen funktionieren wie erwartet, inklusive Klammern und Punkt-vor-Strich:

```
[1] 5
```

```
[1] 1024
```

```
[1] 1
```

Alles hinter einem # ist ein **Kommentar** und wird von R ignoriert. Kommentare sind für Menschen gedacht – nutzen Sie sie großzügig, um sich (und anderen) zu erklären, was ein Befehl tun soll.

Viele Rechenoperationen sind als **Funktionen** verfügbar. Eine Funktion hat einen Namen und bekommt ihre Eingaben in runden Klammern übergeben:

```
[1] 12
```

```
[1] 3.14
```

Bei `round()` sehen Sie ein **benanntes Argument** (`digits = 2`). Solche Argumente steuern das Verhalten einer Funktion, ohne dass Sie sich die Reihenfolge merken müssen.

A.5 - Objekte und Zuweisungen

Ergebnisse können Sie in **Objekten** (Variablen) speichern, um mit ihnen weiterzurechnen. Die Zuweisung geschieht mit dem Pfeil `<-` (in RStudio praktisch über die Tastenkombination `Alt + -`):

```
[1] 15
```

Wenn Sie einem Objekt etwas zuweisen, gibt R zunächst nichts aus – das Ergebnis liegt jetzt „unsichtbar“, im Environment. Tippen Sie den Objektnamen allein ein, wird sein Inhalt angezeigt:

```
[1] 5
```

Ein einzelner Wert ist selten genug. Häufig fassen wir mehrere Werte mit der Funktion `c()` (für *combine*) zu einem **Vektor** zusammen und wenden Funktionen darauf an:

```
[1] 18
```

```
[1] 13.5
```

Damit haben Sie im Grunde schon das Grundprinzip von R gesehen: Werte in Objekten speichern und Funktionen darauf anwenden. Wie man aus einer echten Datei einen solchen Datensatz macht und ihn aufbereitet, ist Thema des Kapitels zum Datenmanagement.

A.6 - Pakete installieren und laden

R bringt von Haus aus schon sehr viele Funktionen mit (das nennt man *base R*). Der eigentliche Reichtum von R steckt aber in **Paketen** – Sammlungen zusätzlicher Funktionen, die andere geschrieben und

über CRAN zur Verfügung gestellt haben. Für Pakete gilt eine einfache Faustregel:

Einmal installieren, in jeder Sitzung laden.

Installiert wird ein Paket mit `install.packages()` (in Anführungszeichen). Das machen Sie nur ein einziges Mal pro Rechner:

Damit R die Funktionen eines Pakets tatsächlich kennt, müssen Sie es zu Beginn jeder Sitzung mit `library()` laden (diesmal ohne Anführungszeichen):

Das **tidyverse** ist eine Sammlung eng verzahnter Pakete für Datenmanagement und Visualisierung, die uns später intensiv begleiten wird. Sie können es also schon jetzt installieren – richtig einsetzen werden wir es in den kommenden Kapiteln.

i Hinweis

`install.packages()` braucht eine Internetverbindung und kann beim ersten Mal etwas dauern. `library()` geht danach sofort.

A.7 - Hilfe bekommen

Niemand kennt alle Funktionen und Argumente auswendig – auch ich nicht. Zu jeder Funktion gibt es eine Hilfeseite, die Sie mit einem vorangestellten Fragezeichen aufrufen:

Die Hilfeseite erscheint dann rechts unten im *Help*-Pane und erklärt, welche Argumente eine Funktion erwartet und was sie zurückgibt. Ganz unten findet sich fast immer ein Abschnitt mit lauffähigen **Beispielen** – die lohnt es sich, einfach einmal auszuprobieren. Und keine Sorge: Fehlermeldungen gehören zum Programmieren dazu. Sie in Ruhe zu lesen (und notfalls zu googeln) ist eine ganz normale, produktive Tätigkeit.

A.8 - Mit KI arbeiten

Neben der eingebauten Hilfe steht Ihnen heute ein weiteres, erstaunlich mächtiges Werkzeug zur Verfügung: **große Sprachmodelle** (engl. *large language models*, LLMs). Sie kennen sie vermutlich als Chatbots wie ChatGPT, Claude oder Gemini. Daneben gibt es eine wachsende Zahl **offener Modelle** – etwa aus den Familien Llama, Mistral, Qwen oder Gemma –, die Sie über die Hochschule, über verschiedene Anbieter oder sogar lokal auf dem eigenen Rechner (z.B. mit *Ollama* oder *LM Studio*) nutzen können. Für R und Statistik sind diese Modelle bemerkenswert nützlich – wenn man weiß, wofür sie taugen und wofür nicht.

Am besten stellen Sie sich ein Sprachmodell als einen sehr belese-
nen, immer geduldigen, aber gelegentlich übermütigen Tutor vor: Es hat unzählige R-Beispiele „gesehen“, und formuliert flüssig – aber es *versteht* nicht wirklich, was es sagt, und irrt sich manchmal mit voller Überzeugung. Genau deshalb ist es ein hervorragendes Hilfsmittel und ein schlechter Ersatz für eigenes Verständnis.

A.8.1 - Wofür Sprachmodelle sich gut eignen

- **Code erklären lassen.** Sie kopieren einen Codeblock hinein und lassen ihn Zeile für Zeile erläutern. Das ist oft schneller verständlich als eine Hilfeseite.
- **Fehlermeldungen entschlüsseln.** R-Fehlermeldungen wirken am Anfang kryptisch. Kopieren Sie die vollständige Meldung samt Ihrem Code, und lassen Sie sich erklären, was schiefgelaufen ist.
- **Einen ersten Entwurf schreiben.** Beschreiben Sie in Worten, was Sie erreichen wollen, und Sie bekommen einen Code-Vorschlag, den Sie ausprobieren und anpassen können.
- **Konzepte am Beispiel erklären.** „Erkläre mir den Unterschied zwischen `mean()` und `median()` an einem kleinen Beispiel, – Sprachmodelle sind gut darin, Abstraktes an konkreten Fällen festzumachen.
- **Zwischen Werkzeugen übersetzen.** Wenn Sie etwas aus Excel oder SPSS kennen, können Sie fragen, wie dasselbe in R (oder im `tidyverse`) aussieht.
- **Code aufräumen und kommentieren.** Lassen Sie sprechende Variablennamen vorschlagen oder Kommentare ergänzen, damit Ihr Skript auch in einem halben Jahr noch lesbar ist.

A.8.2 - Wo Vorsicht geboten ist

- **Erfundene Funktionen.** Modelle „halluzinieren“, gelegentlich Funktionen, Argumente oder ganze Pakete, die es gar nicht gibt – und das völlig plausibel. **Führen Sie vorgeschlagenen Code immer selbst aus** und prüfen Sie, ob er das Erwartete tut.
- **Veraltetes Wissen.** Der Trainingsstand eines Modells liegt in der Vergangenheit, und Pakete ändern sich. Was das Modell vorschlägt, muss nicht der heute übliche Weg sein.
- **Selbstbewusst daneben.** Besonders bei statistischen Fragen kommt manchmal eine sachlich falsche Antwort im Brustton der Überzeugung. Um das zu erkennen, brauchen Sie eigenes Fachwissen – genau das, was dieser Kurs aufbaut.
- **Datenschutz.** Geben Sie keine sensiblen oder personenbezogenen Forschungsdaten in fremde Cloud-Dienste ein. Hier spielen offene, lokal laufende Modelle ihren großen Vorteil aus: Die Daten verlassen Ihren Rechner nicht.
- **Das Verständnis nicht auslagern.** Wenn das Modell alles schreibt, lernen Sie nichts. Nutzen Sie es, um zu *verstehen*, nicht um das Verstehen zu vermeiden.

A.8.3 - Gute Prompts

Wie hilfreich die Antwort ist, hängt stark davon ab, wie Sie fragen. Drei Faustregeln:

1. **Kontext geben.** Sagen Sie, dass Sie Anfänger:in sind und mit R/ RStudio (ggf. dem `tidyverse`) arbeiten.
2. **Konkret werden.** Fügen Sie Ihren echten Code, die vollständige Fehlermeldung oder einen kleinen Ausschnitt Ihrer Daten bei.
3. **Erklärung verlangen, nicht nur Lösung.** Bitten Sie darum, den Weg zu erläutern – so lernen Sie etwas dabei, statt nur zu kopieren.

Ein paar Formulierungen, die sich bewährt haben (einfach kopieren und anpassen):

Ich bin R-Anfänger:in und arbeite in RStudio. Erkläre mir bitte diesen Code Zeile für Zeile in einfachen Worten:

```
werte <- c(4, 8, 15, 16, 23, 42)
mean(werte)
```

Ich bekomme in R diese Fehlermeldung. Was bedeutet sie, und wie behebe ich sie? Bitte erkläre mir die Ursache, nicht nur die Lösung.

[hier Ihren Code und die vollständige Fehlermeldung einfügen]

Ich möchte in R den Mittelwert und die Standardabweichung für die Zahlen 4, 8, 15, 16, 23, 42 berechnen. Schreibe mir den Code und erkläre kurz, was jede Zeile tut.

Erkläre mir den Unterschied zwischen Standardabweichung und Standardfehler an einem kleinen, anschaulichen Beispiel – so, als würdest du es einer Studienanfängerin erklären.

Tipp

Eine besonders lehrreiche Bitte: „Gib mir zwei Lösungswege und erkläre die Vor- und Nachteile.“ So sehen Sie, dass es selten nur einen richtigen Weg gibt – und trainieren genau das kritische Urteil, um das es in diesem Kurs geht.

Damit sind die Vorbereitungen abgeschlossen. Im nächsten Kapitel steigen wir in das Datenmanagement ein – und arbeiten zum ersten Mal mit echten Daten.